91602
new 82-10

ontario hydro
research division

COMPLIMENTARY COPY
Please Credit Ontario Hydro Research if
Reproduced in Whole or in Part

SENSITIVITY STUDY OF THE CONTINUOUS RELEASE DISPERSION MODEL (CRDM) FOR RADIOACTIVE POLLUTANTS

Report No 87-114-K

F. Camacho
Mathematician
Statistics & Reliability Section
Operations Research Department

ABSTRACT

The Continuous Release Dispersion Model (CRDM) is used to calculate spatial distribution of pollutants and their radiation doses in the event of accidental releases of radioactive material from Nuclear Generation Stations. A sensitivity analysis of the CRDM was carried out to develop a method for quantifying the expected output uncertainty due to inaccuracies and uncertainties in the input values. A simulation approach was used to explore the behaviour of the sensitivity functions. It was found that the most sensitive variable is wind speed, the least sensitive is the ambient temperature, and that largest values of normalized concentrations are likely to occur for small values of wind speed and highly stable atmospheric conditions. It was also shown that an error between 10% and 25% should be expected in the output values for a 1% overall error in the input values, and this factor could be much larger in certain situations.

job	file	date	report no.
35-618-001	837.07	August 4, 1987	87-114-K

TABLE OF CONTENTS

	Page
1. INTRODUCTION	1
2. CONTINUOUS RELEASE DISPERSION MODEL (CRDM)	1
3. THE SENSITIVITY FUNCTION	2
3.1 Sensitivity and Partial Sensitivity Functions	2
3.2 Relative Sensitivity Functions	4
4. SENSITIVITY ANALYSIS OF CRDM	4
4.1 Sensitivity Function of CRDM	4
4.2 Simulation of the Input Values	5
4.3 Results and Discussion	6
4.3.1. Sensitivity Functions	6
4.3.2. Normalized Concentrations	7
5. CONCLUSIONS AND RECOMMENDATIONS	8
REFERENCES	10
APPENDIX A - An Analysis of Model Error	A-1
APPENDIX B - Partial Derivatives of the Continuous Release Dispersion Model	B-1
APPENDIX C - Description of the Programs	C-1
DISTRIBUTION	last page

91601
new 82-10

ontario hydro
research division

To Dr. D. Mills
Director of Research

SENSITIVITY STUDY OF THE CONTINUOUS RELEASE DISPERSION MODEL (CRDM) FOR RADIOACTIVE POLLUTANTS

1. INTRODUCTION

A sensitivity analysis of the CRDM was carried out by the Statistics Unit in response to an inquiry by the Environmental Science Section of the Chemical Research Department. The CRDM was developed by the Department for use in the Ontario Nuclear Emergency Plan. In the event of accidental releases of radioactive material from Nuclear Generation Stations (NGS), the model calculates projections of the spatial distribution of pollutants and the radiation doses expected within the exposure area given predicted source terms and meteorological conditions.

The sensitivity analysis is a concept long used in system studies to determine the variation in the behavior of a system caused by small changes in the values of its parameters. In this report the sensitivity concept is used in a slightly different way, namely, to develop a method for quantifying the expected variation in the output of the CRDM caused by uncertainties and inaccuracies in the input values. This quantification is very useful when the model is used in a decision making process, and is a necessity if the model is of a predictive nature, as the CRDM is.

2. CONTINUOUS RELEASE DISPERSION MODEL (CRDM)

The Nuclear Energy Atmospheric Dispersion and Dose Model (NEADM) described in detail in /1, 2 and 3/ is composed of a Gaussian Puff Model (AGPUF) used to estimate the instantaneous concentrations and exposures due to short duration releases, and a Continuous Release Dispersion Model (CRDM), used to calculate concentrations and deposition amounts due to long duration releases.

In this study only the CRDM component is considered. A diagram of CRDM is presented in Figure 1 (a complete description can be found in references /2, 3/). As can be seen from this figure, the CRDM model is made up of two components. The first, a diffusion model (summarized in Appendix B), calculates the normalized concentration or dilution factor (i.e. the concentrations of contaminants per release rate) based on site and meteorological information. The second, calculates the actual concentrations and deposition densities based on the normalized concentrations and the source term information.

For the sensitivity study, it is sufficient to carry out the analysis for the diffusion model because the normalized concentrations summarize the effects of all the input variables except the information of the source term. In this way the results will be independent of the type of radionuclide and could be applied to any pollutant.

3. THE SENSITIVITY FUNCTION

This section presents the theoretical definitions of the general, the partial, and the relative sensitivity functions. These concepts are adaptations of those given in/4/ and will be used in the rest of the report.

3.1 Sensitivity and Partial Sensitivity Functions

The general idea of a sensitivity function is to quantify the variation of the behavior of a system with respect to the variation of certain parameters of interest. This report concentrates in an input sensitivity analysis, and for that reason, the input values are assumed to be the parameters of interest. A formal definition of sensitivity can be stated as follows.

- Suppose that the system can be expressed by the model

  $$Y = f(\underline{X})$$

  where,

  Y : represents the output of the system (in the given case, the normalized concentrations)

  $\underline{X}$: $(x_1, \ldots, x_n)$ represents the vector of input variables (the site and meteorological information)

  f(.) : represents the set of equations which calculate the normalized concentrations for the given input vector $\underline{X}$ (the equations given in Appendix B).

  It is assumed that f(.) is exact (i.e. if the true value of $\underline{X}$ is given, then $f(\underline{X})$ will compute the exact value of Y). This, of course, is not completely true, as is explained in Appendix A, but the assumption does not significantly change the results.

- Suppose that an error, $\Delta\underline{X}$, is made in the specification of the input values. Denote by $\underline{X}_o$ the given input vector, and by $Y_o = f(\underline{X}_o)$ the calculated value of the normalized concentrations. Denote the true input value as $\underline{X} = \underline{X}_o + \Delta\underline{X}$, with corresponding output value $Y = f(\underline{X})$. The output error incurred by incorrectly specifying the value of the input parameters is then given by

  $$\Delta Y = Y - Y_o.$$

- The general sensitivity function is defined as the function which calculates the magnitude of ΔY in terms of $\underline{X}_o$ and $\Delta\underline{X}$. When the errors in the input values are small (i.e., when $\Delta X_i << X_{oi}$), the sensitivity function can be expressed as a linear function which relates the elements of $\Delta\underline{X}$ to the output error ΔY in the form (see/4, Chapter 2),

  $$\Delta Y = S(\underline{X}_o) \cdot \Delta\underline{X}.$$

In the case that $f(\underline{X})$ can be assumed to be a derivable function almost everywhere and the

second order factors in a Taylor's expansion can be neglected, then the sensitivity function can be approximated by:

$$S(\underline{X}_o) = \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n}\right). \qquad (1)$$

The output error can then be approximated by:

$$\Delta Y = \sum_{i=1}^{n} \frac{\partial f}{\partial x_i} \Delta X_i.$$

The partial derivatives $\partial f/\partial x_i$, i = 1, ..., n, are also known as the partial sensitivity functions.

The partial sensitivity functions can be used to quantify the output errors associated with uncertainties and inaccuracies in the input values when a quantification of the input accuracy is known. For example, suppose that the errors Δx_i, i=1,...,n can be assumed to be independent random variables with zero mean and standard deviation s_i; then the error variance of the output can be estimated by

$$S_y^2 = \sum_{i=1}^{n} \left(\frac{\partial f}{\partial x_i}\right)^2 S_i^2.$$

This variance can be used to obtain an approximate p% confidence interval for the output using the expression:

$$Y = Y_o \pm Z_{p/2} \cdot S_y$$

were $Z_{p/2}$ represents the p/2 percentile of the normal distribution.

3.2 Relative Sensitivity Functions

In many cases it is of interest to detect which are the most sensitive input variables, i.e., those variables for which a small variation in the given value will produce the largest variation in the output value. To detect these variables, it is necessary to compare the partial sensitivity function with respect to one variable, for example the wind speed, with the sensitivity function with respect to another variable, for example the ambient temperature. It may seem that a direct comparison of the partial derivatives could be enough. However, these derivatives are all expressed in different units and therefore such an approach would be incorrect.

To carry out the comparison, the relative partial sensitivity functions are introduced. For each input variable x_i, the relative partial sensitivity function is defined as:

$$\frac{\partial \log f(X)}{\partial \log x_i} = \frac{X_i}{f(\underline{X})} \frac{\partial f(X)}{\partial x_i}. \qquad (2)$$

This function can be interpreted as follows. Suppose that there is a p% error in the input variable x_i whereas the other variables are assumed to be correct. Then the percentage error of the output variable is given by $(\partial \log f(\underline{X})/\partial \log x_i)p\%$. Observe that the relative sensitivities are dimensionless quantities and therefore, a direct comparison of these functions can easily be done.

Using the relative sensitivities, the sensitivity function (Equation 1) can be calculated as:

$$S(\underline{X}_o) = f(\underline{X}_o) \left(\frac{\partial \log f(X)}{\partial \log(x_i)} , \ldots , \frac{\partial \log f(X)}{\partial \log(x_n)} \right).$$

When the input errors Δx_i, i=1,...,n, are expressed in percentages, p_i, i=1,...,n, respectively, the associated percentage change in Y can be easily calculated with the relative sensitivities as

$$\Delta Y = \sum_{i=1}^{n} \frac{\partial \log f(X)}{\partial \log(x_i)} p_i.$$

An upper bound for the relative error of Y associated with an overall 1% error in the input variables can be calculated as the sum of the absolute values of the relative derivatives. This value, referred hereafter as the "Total Percent Sensitivity" gives an idea of the overall sensitivity of the system to the input values.

4. SENSITIVITY ANALYSIS OF THE CRDM

This section presents the input sensitivity analysis of the CRDM. The study is carried out to quantify the expected inaccuracies of the predictions of the normalized concentrations due to errors in the values of the input variables.

4.1 Sensitivity Function of the CRDM

In this section, the sensitivity function of the diffusion model is presented. As shown in Section 3.1, the sensitivity function is composed of partial derivatives of the function $f(\underline{X})$ with respect to the input variables $x_1, \ldots, x_n$. In the given case, this function is composed of the the partial derivatives of the normalized concentrations with respect to the following variables:

- wind speed,
- ambient temperature,
- heat flux,
- surface roughness,
- wind measurement height,
- downwind distance,
- crosswind distance,
- height,
- standard deviation of wind direction.

The remaining input variables, namely, date, time, cloud cover, and the latitude of the source, are summarized in the "heat-flux" variable. Analytical expressions of these partial derivatives are given in Appendix B. A computer program written in FORTRAN was implemented to carry out the calculations. The program is described in Appendix C.

To evaluate the partial derivatives it was assumed that the input variables can vary independently. Although this assumption is not completely correct, it provides an adequate working hypothesis which produces approximate results. Because, the dependencies among the input variables are not sufficiently understood, it was not possible to formulate an alternate model which could be used for a more rigorous analysis.

4.2 Simulation of the Input Values

As discussed in Appendix B, the mathematical expressions for the derivatives of the dispersion model are rather complicated, which makes it impossible to study the sensitivity functions analytically. Therefore, it was necessary to take a numerical approach to study the behaviour of the sensitivities.

In a numerical approach, the sensitivity functions are evaluated at a set of points which sufficiently resemble the sample space of the input variables. Then, the frequency distributions of these numerical values are studied to explore the behavior of the sensitivity functions.

To obtain the set of points at which the sensitivity function was evaluated, it was necessary to implement the following simulation procedure (the subroutine SIMVAL, listed in Appendix C, was used to carry out the calculations).

- First, a tape of data containing historical meteorological records at Pickering was obtained from the Environmental Studies and Assessments Department of Ontario Hydro. This tape contains hourly information of wind speed, wind direction, stability class, and precipitation values for a period of five years (1980-1984).

- Second, a random sample of such records was taken to generate the values of Date (yymmdd), time and wind speed. The values of stability class read from the tape were used to generate the values of cloud cover and sigma-theta. The values of cloud cover were obtained using a method given by Pasquill/5/. The incoming solar radiation values necessary to apply the Pasquill's method were obtained by randomly selecting a point within an interval defined by the values of stability class and wind speed. The values of sigma-theta were selected as the middle point of the sigma-theta interval corresponding to the Pasquill Categories as given in/6/, Table 2.3.3.

- Third, the values of temperature were obtained by taken the average historical value of the month at Port Hope/7/. The values of surface roughness were randomly selected within the interval [0,1]. The values of downwind location were randomly selected within the interval [0,1000 m]. The values of wind measurement height were fixed at 10 m and the source latitude at 45° north.

The advantage of using a simulation scheme to generate the input values is that it provides an adequate set of input values to explore the distribution of the output values and of the sensitivity functions. Such distributions identify when the dispersion function is likely to be more sensitive to errors in the input values, and how often this occurs.

Another important by-product of the simulation is that the behaviour of the normalized concentrations can be studied. These analyses can be quite useful. For example, favorable conditions for releasing pollutants in emergency situations can be identified by a perusal of the joint distributions of the normalized concentrations and the input variables.

4.3 Results and Discussion

4.3.1 Sensitivity Functions

This section presents the frequency distributions of the sensitivity functions of the CRDM and a summary of the behaviour of the normalized contaminations with respect to the input values. The frequency distributions were based on 1390 records randomly selected out of the 43848 available on the historical tape. The random numbers were generated using the procedure outlined in/8/ and/9/. Although 2500 numbers were generated, the sample was initially reduced to 2421 because there were 79 numbers with the same value. The effective number of points was further reduced to 1390 after omitting from the sample the records with a wind speed value less than 2.7 m/s. This wind speed value represents the minimum value at which CRDM is applicable/10/.

The frequency distributions are efficiently summarized using box- and-whisker plots /11/. In a box-and-whisker plot, a thin rectangle (box) crossed with a bar is initially drawn. The box represents the observations located between the upper and the lower quartiles, and the cross bar indicates the location of the median of the distribution (the quartiles and the median are the points which divide the set of observations into four groups, each one containing 25% of the observations). Upper and a lower "whiskers" are then attached to the extremes of the box. The whiskers are drawn from the edge of the box to either the corresponding extreme value or the point located at a distance equal to 1.5 times the length of the box, whichever is the closest (the extreme values are the maximum and the minimum of the distribution). Finally, the points lying beyond the whiskers, "far out" values, are added to the plot, indicated as dots.

Figure 2 presents box-and-whisker plots of the relative derivatives of the CRDM with respect to six variables, namely, wind speed, temperature, heat flux, surface roughness, wind measurement height and down wind distance, all plotted in the same scale to facilitate comparisons. The derivatives were calculated using the software described in Appendix C. It can be seen from the plot that, in general, the partial relative sensitivities have skewed distributions, with 75% of the values located around zero and the remaining 25% spread towards either positive or negative regions. It can also be seen that the variable showing the smallest sensitivity is temperature and the variable with the largest sensitivity is wind speed.

These results indicate that most of the time the output values are not very sensitive to changes on the input values. However, there is a 25% chance to under or over estimate the output values, depending on whether the input values are under or over estimated. When the sensitivities spread over the positive region, it indicates that if the input values are underestimated, the output values are also underestimated and vice versa. If the sensitivities spread over the negative region, it indicates that if the input values are underestimated, the output values are overestimated and vice versa.

Figure 3 gives a scatter plot of the values of the Total Percent Derivatives, TDXQ, versus the values of the normalized concentrations (restricted to the range $[0, 15 \times 10^{-7}]$). Figure 4 shows a frequency histogram of the observations of the TDXQ, where the range of TDXQ has been restricted to the interval [0,25]. The range in these plots were reduced to increase the resolution of the plot at the lower end of the scale. The main observation that can be drawn from these plots is that the value of the total percent sensitivity will lie between 2.5 to 10.0 most of the time, although in some occasions it may be higher and even reach values larger than 100.

Recalling what was said in Section 3.2 about the Total Percent Derivatives, the above result implies that for each overall 1% error in the input values, the calculated normalized

concentrations, and therefore, the calculated radiation levels, will have an error whose upper bound lies between 2.5% and 10% of the calculated values. These errors should always be kept in mind in the decision making process. It was observed that the Total Percent Sensitivity can exceed 50%, particularly when the calculated values are small.

Figures 5 to 9 present scatter plots of the normalized concentration versus the variables wind speed, temperature, heat-flux, surface roughness and down-wind distance and their associated relative sensitivities. It can be observed that the general pattern in these plots is that the sensitivity is small at large values of the normalized concentrations and that high values of the sensitivity functions are associated with small values of the normalized concentrations.

4.3.2. Normalized Concentrations

The behaviour of the normalized concentrations with respect to the input variables is now considered. Figures 5 to 9 present plots depicting the general behavior of the normalized concentrations, denoted by "X/Q", with respect to the input variables. Each of these plots can be thought of as the distribution of the values of X/Q with respect to the input variable, and can be interpreted in the following way. Take a fixed value, V_o, of a given input variable. Then, the values of X/Q at V_o give an idea of the expected range of values X/Q may take when the input value of the particular variable is V_o, regardless of the values taken by the other input variables. It is necessary, however, to view these results with some caution when the number of observations associated with V_o is small.

The main results observed on the plots can be summarized as follows.

- For wind speed, it is observed that large values of wind speed produce small values of the normalized concentrations, and that the range of X/Q values tend to decrease as wind speed increases. The range of X/Q for wind speeds greater than 7 m/s is difficult to assess because there were not too many observations in this range as can be seen from the histogram of the values of wind speed given in Figure 10.

- For ambient temperature, it is observed that the range of values of the normalized concentrations remains almost constant over the sample space of the input variable.

- For the heat-flux variable, it is observed that for positive values of the variable, the variation of X/Q is small, whereas there is a tremendous variation on the values of X/Q for values of the heat-flux normally equal to zero i.e. during nighttime. A better understanding of this effect can be obtained by looking at Figure 11, which shows a multiple box-whisker plot of the values of X/Q (calculated at a down wind distance of 1 km) versus the value of sigma-theta, a measure of the atmosphere's stability. From this plot it can be seen how the range and magnitude of the X/Q values increase as the values of sigma-theta decrease. For values of sigma-theta less than 7.5 (corresponding to stability class D and E) the range of X/Q is quite large. It should be pointed out, however, that the distribution of X/Q corresponding to the values of sigma-theta larger than 12.5 may not be very accurate, as is suggested by the distribution of the sigma-theta values given in Figure 12.

- For the surface roughness, Z_o, it is observed from Figure 8 that the range of X/Q is small, although the range seems to be a somewhat larger for smaller values of Z_o.

- For down wind distance, it is observed from Figure 9 that the range of X/Q is larger for small values of the input variable.

5. CONCLUSIONS AND RECOMMENDATIONS

This report presents an input sensitivity study of CRDM where a simulation approach was taken to explore the behavior of the output values and of the sensitivity functions defined in Section 3. The results show that large values of the normalized concentrations are likely to occur for small values of wind speed and during nighttime, ie, under highly stable conditions. It is also shown that large values of the sensitivity function are associated with small values of the normalized concentration and that the most sensitive input variable is wind speed, whereas the least sensitive is the ambient temperature. Finally, it is shown that for every 1% overall error in the input values, an error between 2.5% and 10% should be expected in the output value, and that this factor can be much larger in certain situations.

The sensitivity functions presented in this report can be easily implemented as part of the CRDM in order to give a measure of uncertainty of the calculated concentration. To have an idea of how the sensitivity functions could be implemented, the following example should be considered.

Suppose that a maximum concentration C_{MAX} is not to be exceeded for a projected release. Suppose also, that the values of the input variables can not be exactly specified, but they can be said to be correct within a +/- p% range of corresponding specified values (this is a situation very common in practice; the uncertainty is associated with the method used to measure the input variables and the precision of the instruments used to take the measurements). The normalized concentrations, X/Q, and the "Total Percent Sensitivity" value, TDXQ, can then be calculated using CRDM and the sensitivity programs. According to the results given in Sections 3.2 and 4.3, it can be said that the calculated normalized concentrations are correct within a range equal to r%, where r is the product of p and TDXQ.

If (100 + r)% of the projected X/Q is less than C_{MAX}, then it can be said with confidence that the concentration will not exceed the specified limit. However, if C_{MAX} is within 100% and (100 + r)% of X/Q then it can not be said with confidence that projected concentrations will not exceed the specified limit. Of course, the degree of confidence of the statements will depend on the accuracy of the assessment of the input uncertainties.

A final note of caution. In this study, the sensitivity analysis was used to quantify the uncertainty in the predictions of the CRDM due to uncertainties and inaccuracies of the input values, and not to measure the accuracy of the predictions. As discussed in /10/ and in Appendix A, the accuracy of the model refers to the way the predicted values agree with actual field measurements: the better the agreement, the more accurate the model. Of course, an analysis of the accuracy of the model is what is ultimately desired. However, such model validation can only be done when field measurements become available.

Approved:

J.G. Cassan
Manager
Operations Research Dept

Submitted:

F. Camacho
Mathematician
Reliability and Statistics Section

FC:km

REFERENCES

1. Ogram, G.L., Wright, S.C., Daggupaty, S., and Misra, P.K., (1985). "An Atmospheric Dispersion Modelling System for Nuclear Emergency Response". Proceedings of the 6th Annual Conference of the Canadian Nuclear Society, Ottawa, Ontario, pp 19-33, 1937.

2. Wright, S.C., (1983). "CRDM - A Continuous Release Dispersion Model". Ontario Hydro Research Division Report No. 83-277-K.

3. Wright, S.C., (1983). "Dispersion Modelling System for Exercise Pickering 83 - A User Guide". Ontario Hydro Research Division Report No. 83-278-K.

4. Frank, P.M., (1978). "Introduction to System Sensitivity Theory". Academic Press, New York.

5. Pasquill, F., (1974). "Atmospheric Diffusion Models". 2nd John Wiley, Edition, New York.

6. Ontario Ministry of Environment, Air Research Branch. "Meteorological Data Acquisition System (MADS) Manual."

7. Environment Canada, Atmospheric Environment Service (1981). "Canadian Climates Nationals 1951-1980."

8. Wichmann, B.A., and Hill, I.D., (1982) "Algorithm AS 183. An Efficient and Portable Pseudo-Random Number Generator", Applied Statistics V31 (188-190).

9. McLeod, A.I., (1985). "Remark ASR58. A Remark on the Algorithm AS183", Applied Statistics V34 (198-200).

10. Wrigth, S., (1986). "Dispersion Modelling for Nuclear Emergency Response", Ontario Hydro Research Division Report No. 86-366-K.

11. Tukey, J.W., (1977). "Exploratory Data Analysis", Addison-Wesley, Massachusetts.

12. Lewellen, W.S., and Sykes, R.I., (1985), "A Scientific Critique of Available Methods for Real-Time Simulations of Dispersion", Aeronautical Research Associates of Princeton, Inc. Prepared for the Division of Radiation Programs and Earth Sciences, Office of Nuclear Regulatory Research, document number NUREG/CR-4157, ARAP No. 472 RB.

13. National Working Group on Atmospheric Releases of Radioisotopes. (1983) "Recommendations on Meteorological Monitoring".

14. Tam, Y.T., (1984). "A Study of the Dispersion Climatology of the Pickering Area", Ontario Hydro Research Division Report No. 84-137-K.

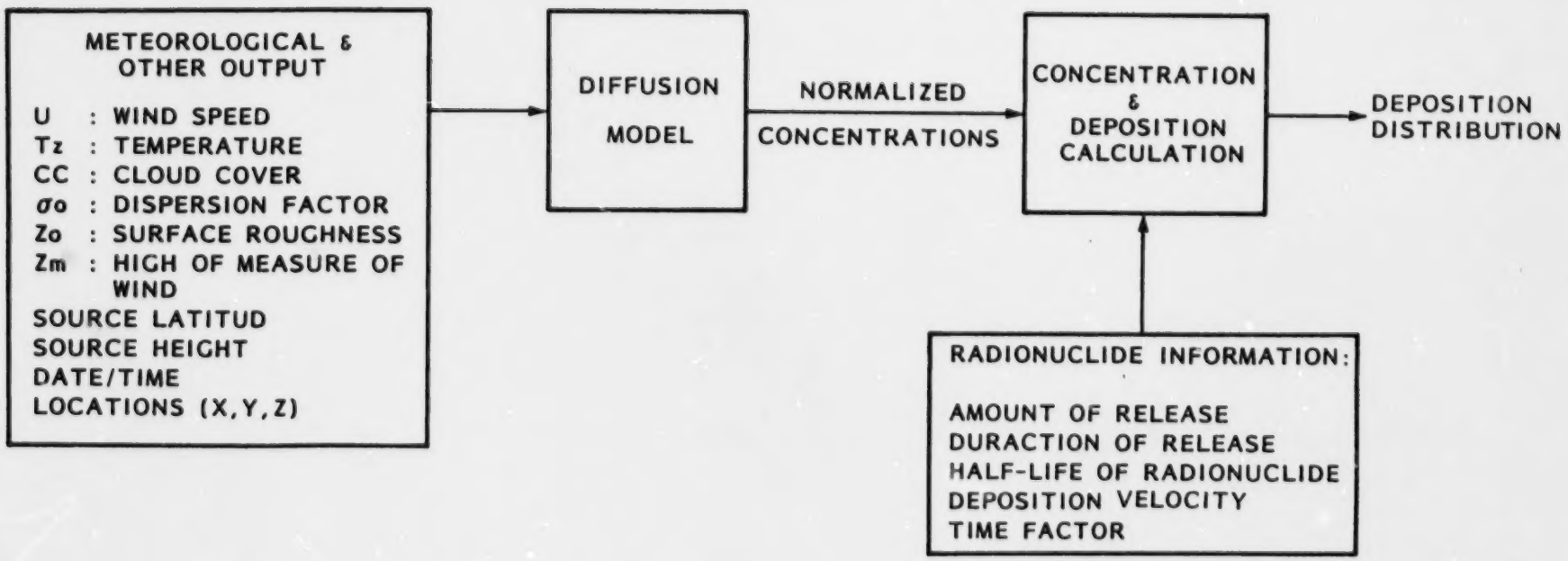

FIGURE 1
THE CRDM MODEL

87-114 171701 RD

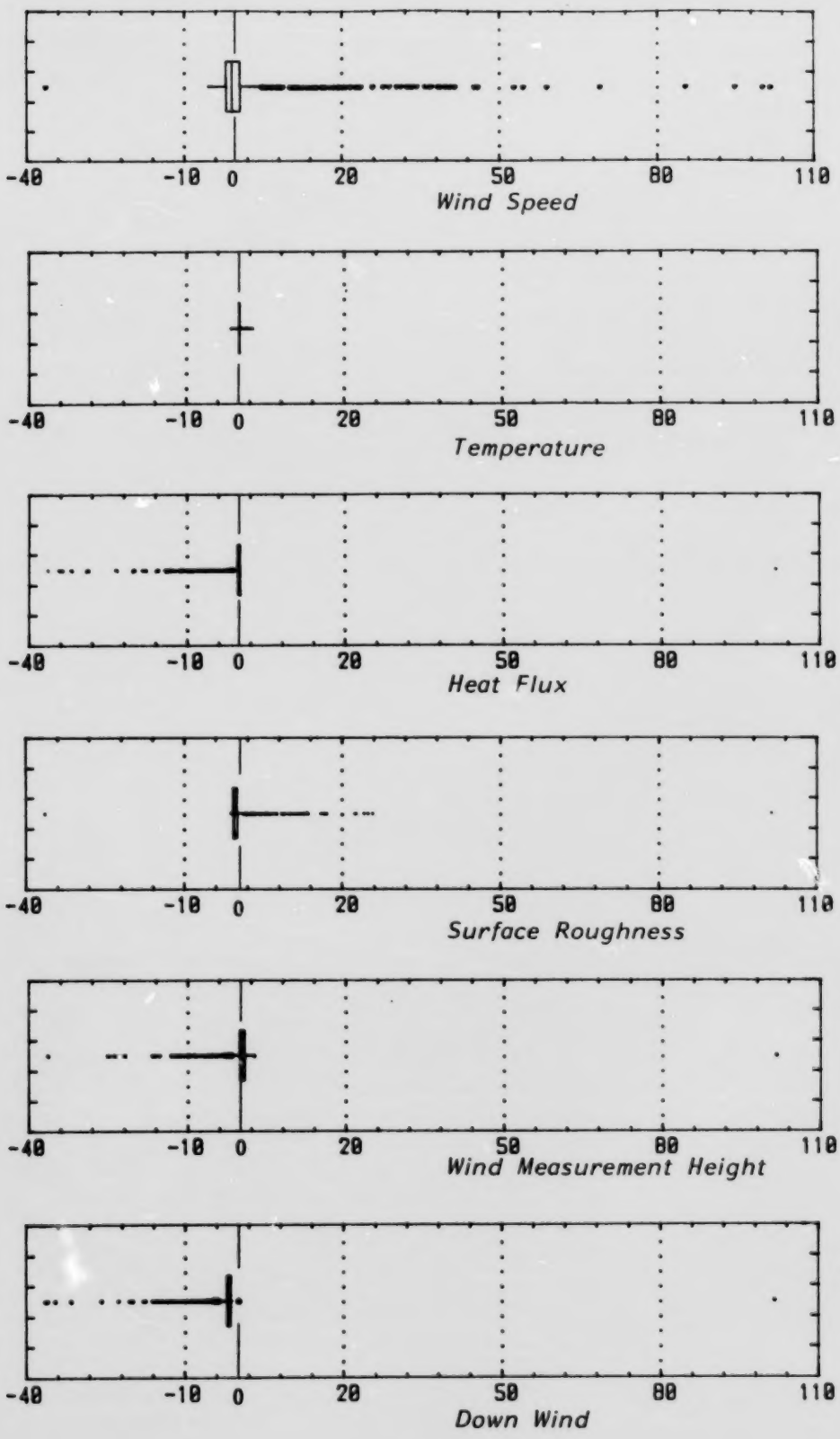

FIGURE 2

BOX-AND-WHISKER PLOTS OF THE RELATIVE DERIVATIVES OF THE CRDM BASED ON 1390 OBSERVATIONS

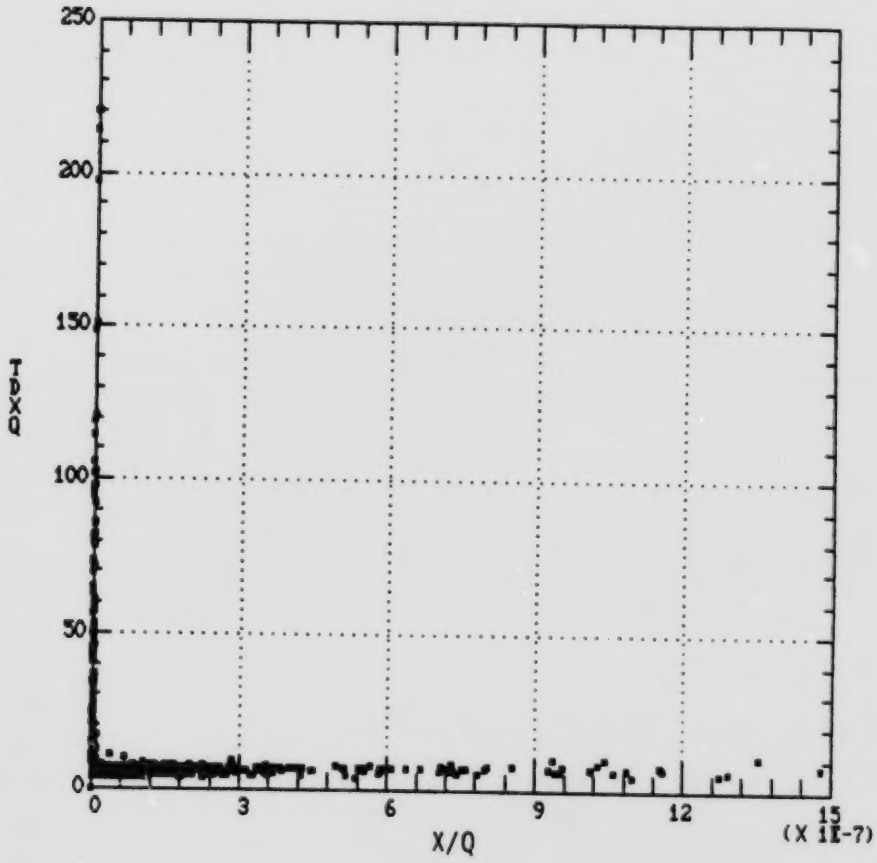

FIGURE 3
SCATTER PLOT
TOTAL PERCENT DERIVATIVES (TDXQ)
VS NORMALIZED CONCENTRATION (X/Q)

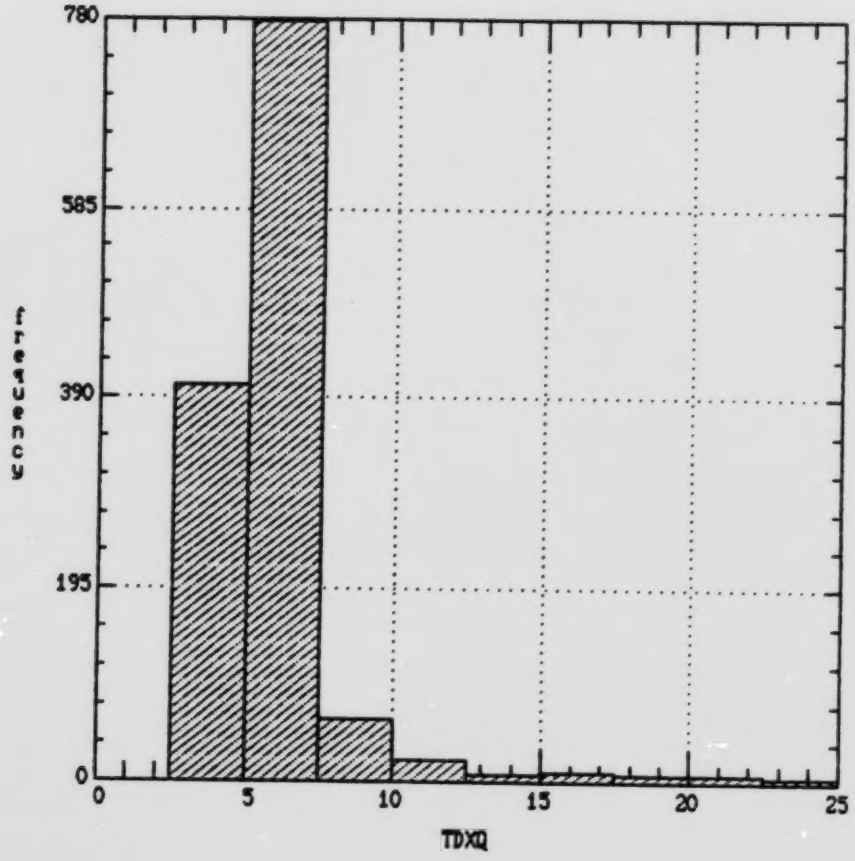

FIGURE 4
HISTOGRAM OF TOTAL PERCENT SENSITIVITY

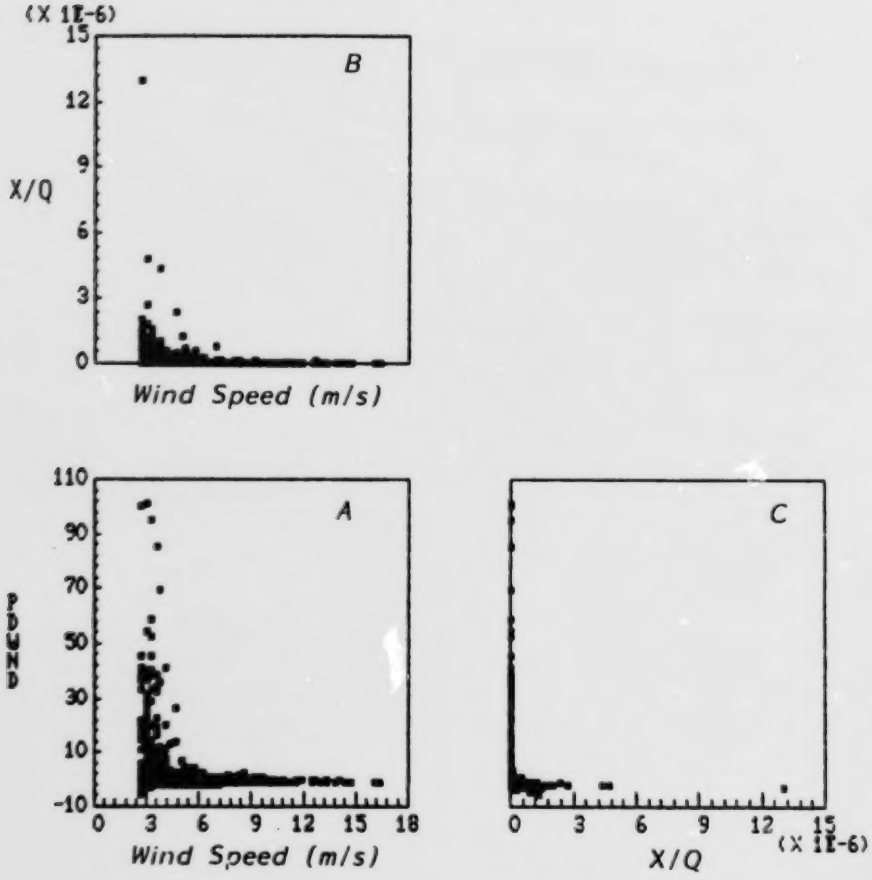

FIGURE 5
SCATTER PLOTS
WIND SPEED

A. Sensitivity Function for Wind Speed (PDWND) vs. Wind Speed

B. Normalized Concentration (X/Q) vs. Wind Speed

C. Sensitivity Function for Wind Speed (PDWND) vs. Normalized Concentration (X/Q)

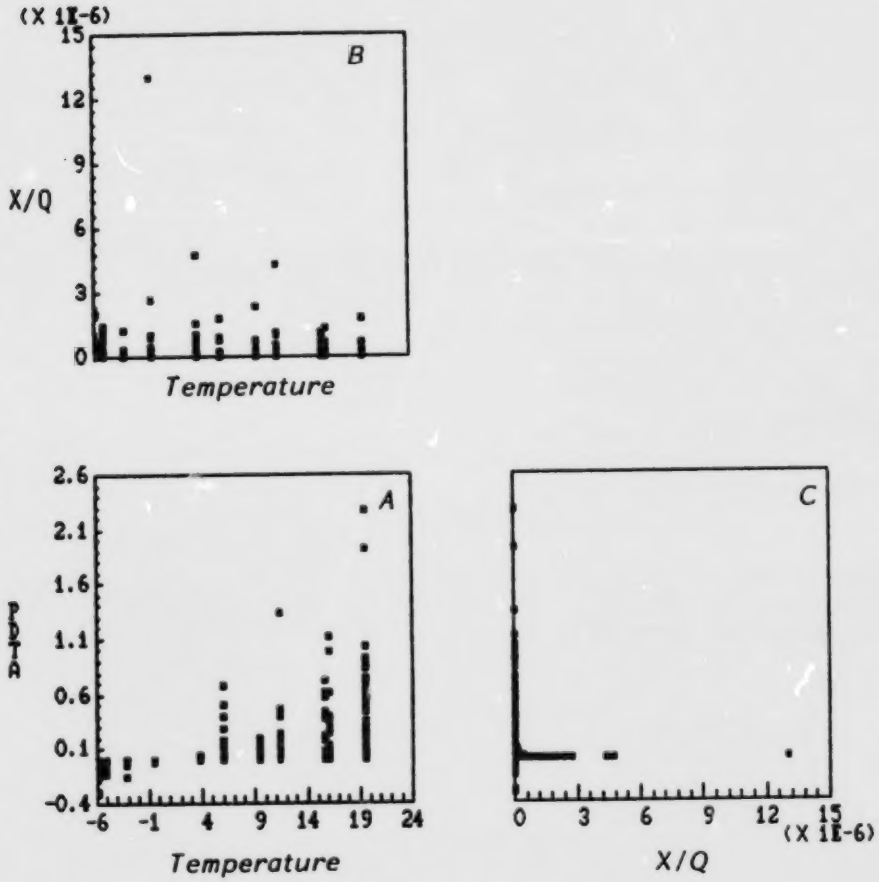

FIGURE 6

TEMPERATURE SCATTER PLOTS

A. Sensitivity Function for Temperature (PDTA) vs. Temperature

B. Normalized Concentration (X/Q) vs. Temperature

C. Sensitivity Function for Temperature (PDTA) vs. Normalized Concentration (X/Q)

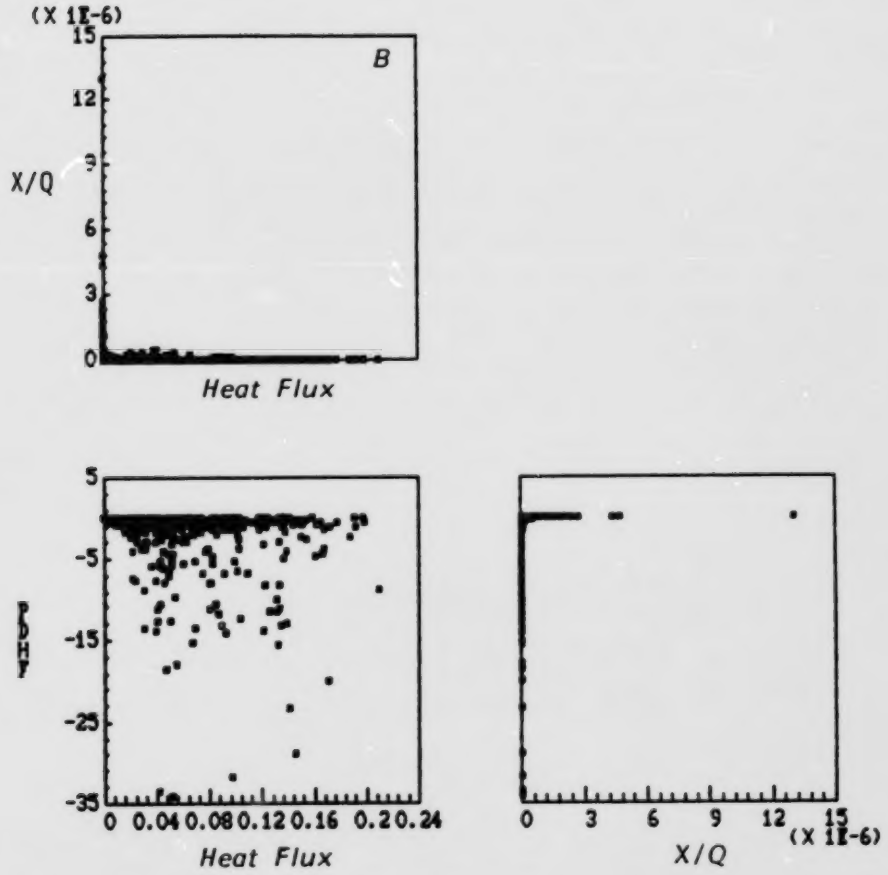

FIGURE 7
HEAT FLUX SCATTER PLOTS

A. Sensitivity Function for Heat Flux (PDHF) vs. Heat Flux

B. Normalized Concentration (X/Q) vs. Heat Flux

C. Sensitivity Function for Heat Flux (PDHF) vs. Normalized Concentration (X/Q)

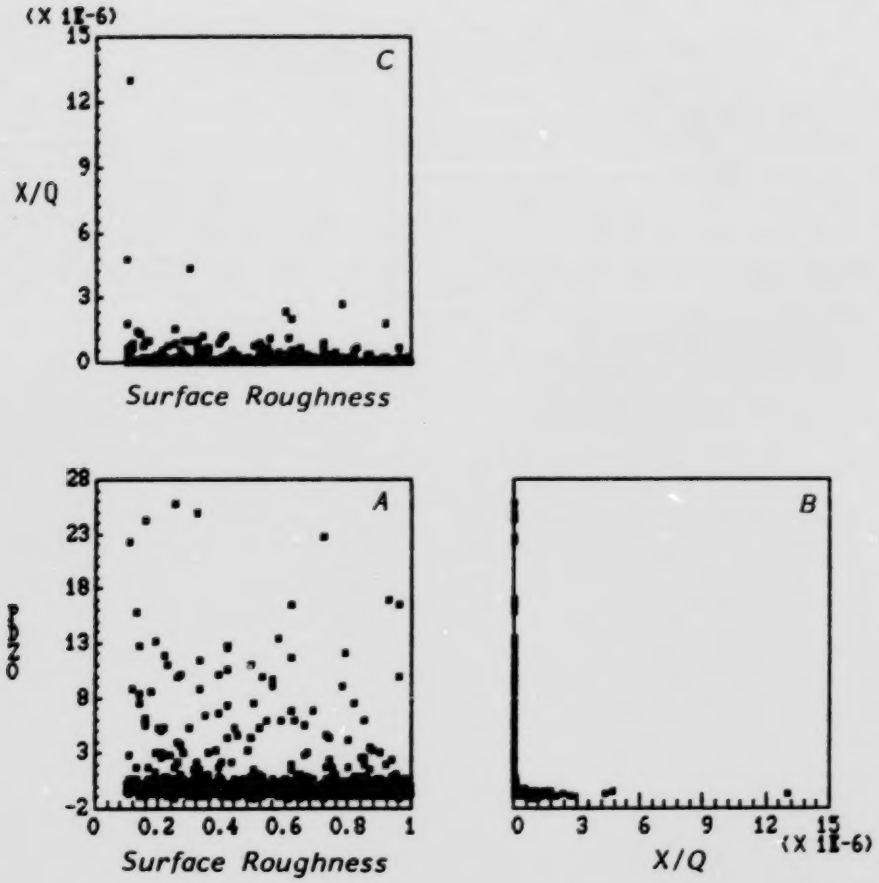

FIGURE 8
SURFACE ROUGHNESS SCATTER PLOTS

A. Sensitivity Function for Surface Roughness (PDZ0) vs. Surface Roughness

B. Normalized Concentration (X/Q) vs. Surface Roughness

C. Sensitivity Function for Surface Roughness (PDZ0) vs. Normalized Concentration (X/Q)

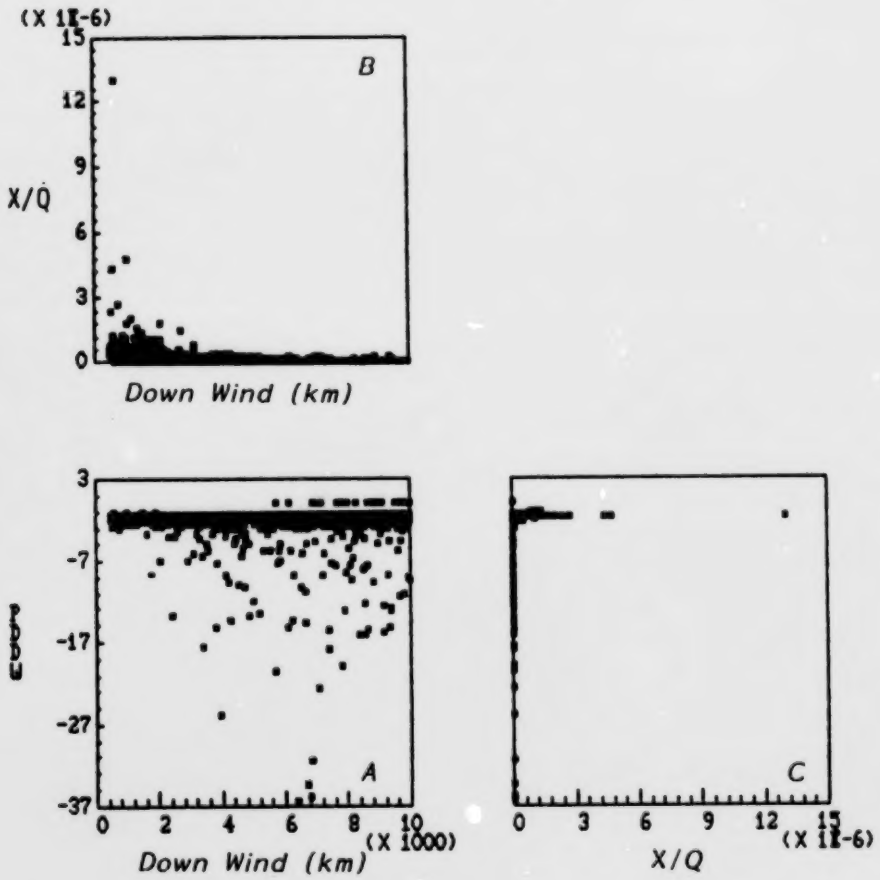

FIGURE 9
DOWN WIND DISTANCE SCATTER PLOTS

A. Sensitivity Function for Down Wind (PDDW) vs. Down Wind

B. Normalized Concentration (X/Q) vs. Down Wind

C. Sensitivity Function for Down Wind (PDDW) vs. Normalized Concentration (X/Q)

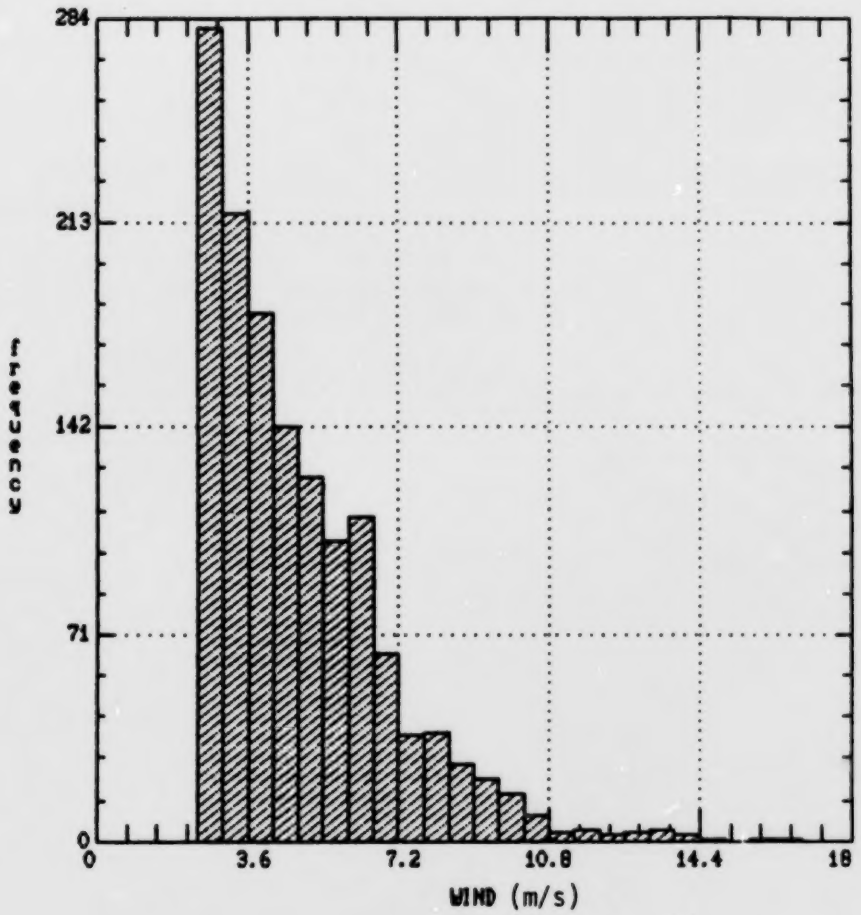

FIGURE 10
HISTOGRAM OF WIND SPEED

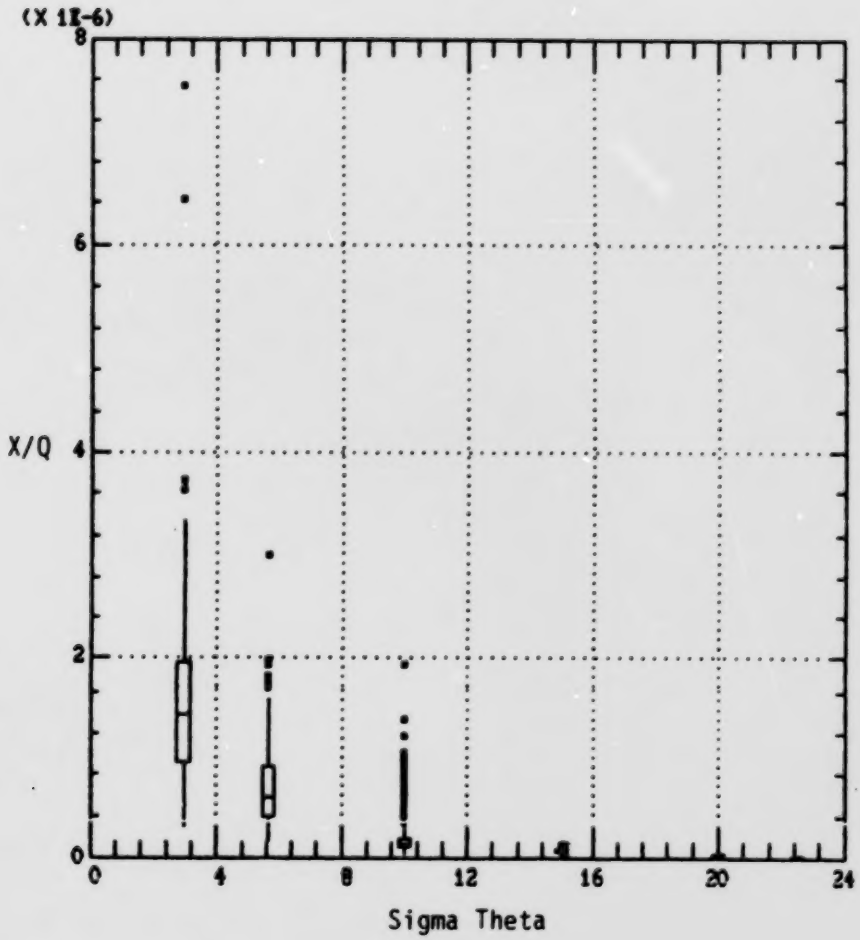

FIGURE 11

MULTIPLE BOX AND WHISKER PLOTS
OF THE NORMALIZED CONCENTRATION (X/Q)
AT DIFFERENT LEVELS OF SIGMA THETA

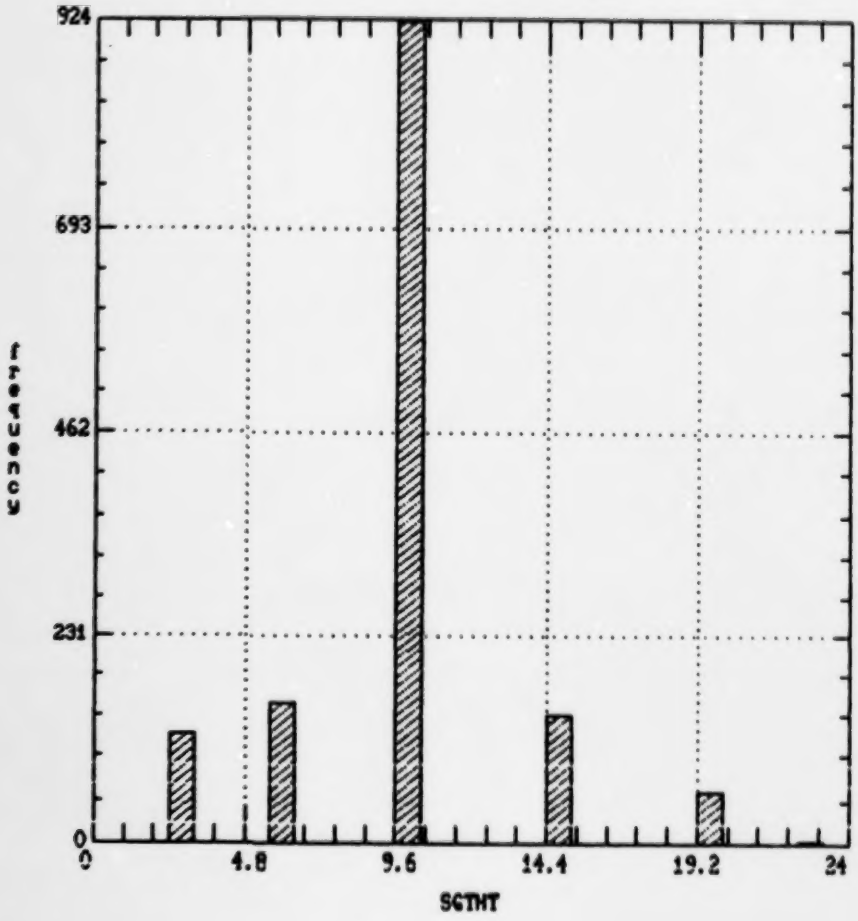

FIGURE 12
HISTOGRAM OF SIGMA THETA

APPENDIX A

AN ANALYSIS OF MODEL ERRORS

TABLE OF CONTENTS

	Page
1. INTRODUCTION	A-1
2. INPUT ERRORS	A-2
3. MODEL ERRORS	A-2
3.1 Specification Errors	A-3
3.2 Calibration Errors	A-3
3.3 Uncertainty Errors	A-3
4. MEASUREMENT ERRORS	A-4
REFERENCE	A-4
FIGURE	A-5

1. INTRODUCTION

This Appendix presents a discussion of model accuracy and tries to give guidelines to identify and correct different sources of errors in a modelling system.

Intuitively, a modelling system is said to be accurate if the predicted values of the model agree sufficiently with actual field observations. There are however difficulties in trying to formalize this definition. First, how do we translate the intuitive notion "sufficiently agree" into a more precise statement? Second, how can we be sure that discrepancies between field measurements and predicted values are indeed due to model inaccuracies and not to errors in the instruments used to obtain the field measurements? Third, what do we consider a modelling system: the set of equations and procedures that take some input values and produce the predicted values, or the aggregate of specified input values, equations, and procedures? Fourth, how could we separate the effects of measurement errors in the input values from the actual inaccuracies of the equations and procedures of the model? We will not attempt to answer all these questions. Rather, a discussion of model errors will be presented hoping that this will throw some light into the problem and will provide ideas for the improvement of the accuracy of the model.

To proceed with the discussion, it is necessary to introduce some notation. Let $Y = f(\underline{\beta}_o)$ denote the predictive value of interest, where f(.) represents the model and $\underline{\beta}_o$ the set of input values. Now, usually the set of input values is only a subset of the general set of conditions $\underline{\beta}$ which determine the real value of the variable, i.e., $\beta = (\underline{\beta}_o, \beta_1)$, say. Let $Z_{\underline{\beta}}$ denote the field observation of the predicted variable corresponding to the set of conditions $\underline{\beta}$. The prediction error $e_{\underline{\beta}}$ is then defined as:

$$e_{\underline{\beta}} = Z_{\underline{\beta}} - Y.$$

Different definitions of model accuracy can be given. For example, if it is desired that all the predictions should be within a certain percentage of the observed values, the model can be said to have a p% accuracy, if for every $\underline{\beta}$ of the possible set of input conditions, the error $e_{\underline{\beta}}$ does not exceed p% of the value of Y, and p is the smallest value for which this is true. If it is not necessary to impose a p% "coverage" for all the predictions, an alternate and perhaps more relevant definition can be obtained by defining p as the expected value, over all the possible input conditions, of the absolute value of the percentage errors, i.e.,

$$p = \int_{\text{All } \underline{\beta}} (e_{\underline{\beta}}/Y)\, F(d\underline{\beta}).$$

Of course, the objective in model development is to obtain a model with the smallest possible value of p.

Several sources of errors which affect the accuracy of the model, or equivalent the prediction error, can then be identified: (i) input errors, associated with $\underline{\beta}_o$; (ii) model errors, associated with f(.); (iii) uncertainty errors, associated with $\underline{\beta}_1$; and (iv) measurement errors, associated with Z. In the following these are explained in some detail.

2. INPUT ERRORS

Input errors are those associated with the inaccuracies in the measurement of and uncertainties associated with the input variables. Different sources are responsible for these errors.

- Inaccuracies in the measurement devices. For example, when the device is not calibrated; in this circumstance, the registered values are incorrect and therefore the input values will contain some error. These errors can be easily controlled and eliminated by adequately maintaining the measurement devices.

- The reading and/or recording is not accurate. In this case, the device may be calibrated but the readings or recordings of the measurement are not correct. These errors can be eliminated using validation and confirmation procedures in the recording of the measurements.

- Stochastic characteristics of the input variable. To illustrate the nature of this error consider the measurement of wind speed. First, the stochastic and dynamic nature of the variable will induce similar characteristics in the output of the system and therefore a comparison of the predicted values with actual field observations is bound to show discrepancies. Second, it is not clear what should be the value of the input variable. For example, is an instantaneous observation of wind speed more representative than a time average? If yes, how is the observation selected? If not, what time interval should be used to take the average? Although this difficulty is not insuperable, the method of measuring the variable is likely to affect the distribution of the prediction error. In any case, it is recommended that a measure of variation should always accompany the actual given values. This measure can be used later to quantify the expected inaccuracy of the output.

Section 3 of the report discusses how the inaccuracies and uncertainties of the prediction errors of CRDM, associated with input errors, can be quantified using an input sensitivity analysis.

3. MODEL ERRORS

Atmospheric dispersion is a very complicated process and generally depend on many variables, some of them quite difficult to identify. Not only may the process depend on many variables, but also the relationships among them may be quite complex. Therefore, any attempt to model the process will convey a good deal of simplification. This is not only practical but also advisable in order to obtain a tractable model. However, this will introduce some inaccuracies in the predictions of the model.

The model errors are by all means the more difficult, if not impossible, to quantify mainly because it requires a complete and correct understanding of the process, which in many situations is not available. Nevertheless, a judgment of the scientific validity of the model will always be obtainable, and this can serve as the basis for further analyses.

The model errors can be classified as follows: i) specification errors; ii) calibration errors; and iii) uncertainty errors.

3.1 Specification Errors

Specification errors are associated with the wrong selection of the relevant variables and/or the wrong specification of the relationships among the variables of the model. Unfortunately, it is very difficult to quantify these type of errors. First, it may not be possible to identify the complete list of variables affecting the system and therefore, it will not be even possible to arrive at the exact form of the model. Second, even when it is possible to identify all the variables of the system, it is still likely that the correct form of the model could not be identified. This makes it difficult to obtain a reference point to estimate the amount of error introduced during the specification of the model. To keep such inaccuracies to a minimum, it is necessary to use as much knowledge of the system as possible and to check the validity of all the assumptions made during the development of the model. Wrong assumptions or incorrect specifications may produce a model which is for all practical purposes useless.

3.2 Calibration Errors

Assuming it is possible to identify all the relevant variables and the correct structure of the process, errors can still be introduced in the model by using wrong values of the parameters of the model. For example, if the system could be adequately modelled by a simple equation like:

$$Y = 3X + 5$$

it is then possible to introduce errors in the model by incorrectly specifying the model as:

$$Y = 3.2X + 5$$

The calibration errors are likely to arise when the parameters of the model are obtained by fitting equations to experimental data, which is quite common in physical sciences. To minimize this type of error, it is necessary to employ adequate statistical techniques and to check the suitability of the data used to estimate the parameters.

3.3 Uncertainty Errors

Uncertainty errors are expected because a model is always a simplification of the system it intends to model, and therefore it can not be expected that the output of the model will exactly agree with the values of the system. In the following, a discussion of the nature of this type of error expected in the CRDM is presented.

First assume that it is possible to identify the complete set of variables of the process and the right structure of the model. Second, assume that the set of variables can be partitioned into two groups, one composed of those included in the model and that can be measured, denoted by β_o, and the other of those which are not included in the model, denoted by β_1. Schematically, the system and the model will look as in Figure A.1 (see also/1/).

Now, for a given observed values of β_o the model predicts the mean value of the output variable, Z_o, over the ensemble of values $\{\beta_o,\beta_1(\beta_o)\}$, where $\beta_1(\beta_o)$ denotes the set of all possible values of β_1 given the measurement β_o. Z_o is usually different from the actual realization of the system, Z, which is determined by the current value (β_o,β_1). If it were possible to measure without error the actual realization of the output of the system over the set of all the possible input conditions $\{\beta_o,\beta_1(\beta_o)\}$, the mean measured value would be equal to Z_o. Therefore, even in this optimal case, any comparison between the predicted and the observed values will always

show a discrepancy. This discrepancy can be expressed as:

$$Z_o = Z + \text{error}(\underline{\beta}_o)$$

i.e. the observed value will always be the predicted value plus some error which may depend on the value of $\underline{\beta}_o$. The error term is a random variable whose distribution is unknown. If this distribution can be obtained, then a measurement of the accuracy of the model could be calculated.

4. MEASUREMENT ERRORS

These errors are those associated with the actual field measurements of the output of the system. Of course if these measurements are not correct, even an exact prediction model will be considered inaccurate. However, it should be pointed out that in any measurement procedure some measurement error is always expected, but this should be small, if the observations were to be of any value.

REFERENCE

1. National Working Group on Atmospheric Releases of Radioisotopes (1983). "Recommendations on Meteorological Monitoring".

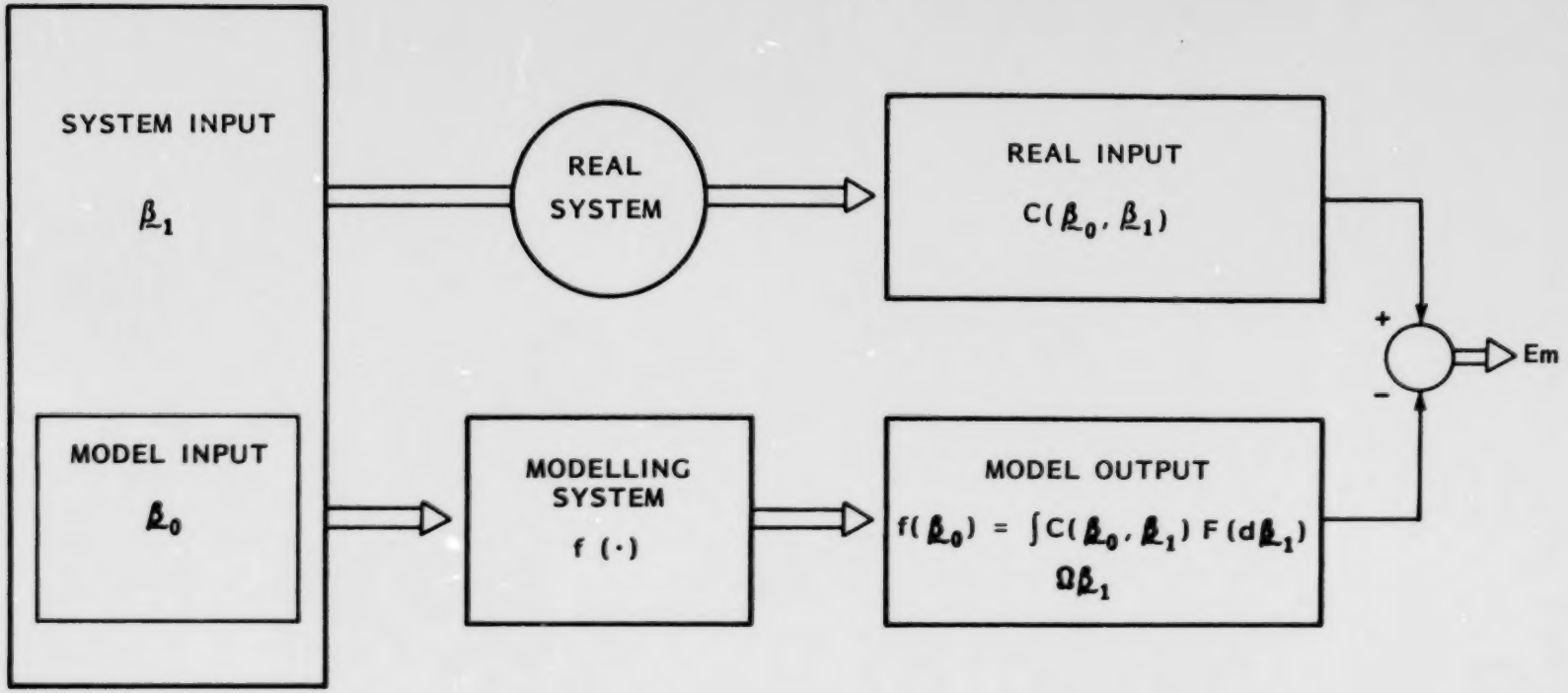

FIGURE A.1
REAL SYSTEM VS. MODEL

APPENDIX B

PARTIAL DERIVATIVES OF THE CONTINUOUS RELEASE DISPERSION MODEL

This Appendix presents the mathematical expressions of the partial derivatives of the Continuous Release Dispersion Model (CRDM) with respect to the input variables. The partial derivatives are used in the calculation of the sensitivity functions (see section 3 and 4 of the report). To facilitate the notation, the letter t will be used to denote any of the input variables.

TABLE OF CONTENTS

		Page
1.	DIFFUSION MODEL	B-2
2.	MEAN HORIZONTAL VELOCITY	B-3
3.	MEAN PARTICLE HEIGHT	B-5
4.	MONIN-OBUKHOV LENGTH	B-7
5.	FRICTION VELOCITY	B-9
6.	SURFACE HEAT FLUX	B-10
7.	SIGMA Y (σ)	B-12

1. Diffusion Model

The diffusion of passive contaminants in the surface layer from sources near the ground can be described by the following equation/3,4,8/:

$$\frac{\chi}{Q} = \frac{0.73}{\sqrt{2\pi}\ \bar{u}\ \bar{z}\ \sigma_y} \exp\left[- \left(\frac{0.66z}{\bar{z}} \right)^{1.5} - \frac{y^2}{2\sigma_y^2} \right]$$

where

χ is the concentration (units.m^{-3})

Q is the emission rate (units.s^{-1})

$\bar{u}$ is the mean horizontal velocity ($m.s^{-1}$)

$\bar{z}$ is the mean particle height (m)

σ_y is the crosswind standard deviation (m)

z is the vertical elevation from the ground (m)

y is the crosswind distance (m)

The derivative of the normalized concentration χ/Q with respect to τ is:

$$\frac{\partial\ (\chi/Q)}{\partial \tau} = - \frac{\chi}{Q} \left\{ \frac{1}{\bar{u}} \frac{\partial \bar{u}}{\partial \tau} + \left(1 - 1.5 \left(\frac{0.66z}{\bar{z}} \right)^{1.5} \right) \frac{1}{\bar{z}} \frac{\partial \bar{z}}{\partial \tau} + 1.5 \left(\frac{0.66z}{z} \right)^{1.5} \frac{1}{z} \frac{\partial \bar{z}}{\partial \tau} + \left(1 - \frac{y^2}{\sigma_y^2} \right) \frac{1}{\sigma_y} \frac{\partial \sigma_y}{\partial \tau} + \frac{y}{\sigma_y^2} \frac{\partial y}{\partial \tau} \right\}$$

2. Mean Horizontal Velocity

The mean horizontal velocity ($\bar{u}$) is given by:

$$\bar{u} = \begin{cases} \frac{u^*}{k}\left[\ln\left(0.6\ \frac{\bar{z}}{z_o}\right) - \psi\left(0.6\ \frac{\bar{z}}{L}\right)\right] & \text{for } \frac{\bar{z}}{L} < 0 \text{ (unstable)} \\ \frac{u^*}{k}\left[\ln\left(0.6\ \frac{\bar{z}}{z_o}\right) + 4.7\ \frac{\bar{z}}{L}\right] & \text{for } \frac{\bar{z}}{L} \geq 0 \text{ (stable)} \end{cases}$$

where

u^* is the friction velocity ($m.s^{-1}$)

z_o is the surface roughness (m)

L is the Monin-Obukhov length (m)

k is the von Karman constant

$$\psi = 2 \ln\left[\frac{(1+a)}{2}\right] + \ln\left[\frac{(1+a^2)}{2}\right] - 2\tan^{-1}(a) + \frac{\pi}{2}$$

$$a = (1-15(0.6\ \frac{\bar{z}}{L}))^{1/4}$$

The derivatives of $\bar{u}$ with respect to τ are given below.

For $\frac{\bar{z}}{L} < 0$

$$\frac{\partial \bar{u}}{\partial \tau} = \frac{\bar{u}}{u^*}\frac{\partial u^*}{\partial \tau} + \frac{u^*}{\bar{z}k}\frac{\partial \bar{z}}{\partial \tau} - \frac{u^*}{k}\left[0.6\ \psi - 2.7\ (\frac{\bar{z}}{L})\right.$$

$$\left.(\frac{1}{1+a} + \frac{a-1}{1+a^2})a^{-3}\right]\ \frac{\partial(\bar{z}/L)}{\partial \tau}$$

For $\frac{\bar{z}}{L} > 0$

$$\frac{\partial \bar{u}}{\partial \tau} = \frac{\bar{u}}{u^*}\frac{\partial u^*}{\partial \tau} + \frac{u^*}{\bar{z}k}\frac{\partial \bar{z}}{\partial \tau} + 4.7\ \frac{\partial(\bar{z}/L)}{\partial \tau}$$

where

$$\frac{\partial(\bar{z}/L)}{\partial\tau} = \frac{1}{L}\left(\frac{\partial\bar{z}}{\partial\tau} - \frac{\bar{z}}{L}\frac{\partial L}{\partial\tau}\right)$$

3. Mean Particle Height

The mean particle height ($\bar{z}$) is solved from:

$$x = \begin{cases} \frac{0.74}{k^2}\ \bar{z}\left\{\left[\ln\left(0.6\ \frac{\bar{z}}{z_o}\right) - \psi\left(0.6\ \frac{\bar{z}}{L}\right)\right]\left[\left(1-3.5\ \frac{\bar{z}}{L}\right)^{-0.5}\right]\right\} & \text{for } \frac{\bar{z}}{L} < 0 \\ \frac{0.74}{k^2}\ \bar{z}\left\{\left[\ln\left(0.6\ \frac{\bar{z}}{z_o}\right) + 4.9\ \frac{\bar{z}}{L}\ \ (1] + 4.9\ \frac{\bar{z}}{L}\right) + 1.2\ \frac{z}{L}\right\} & \text{for } \frac{\bar{z}}{L} \geq 0 \end{cases}$$

where x is the downwind distance (m)

The derivatives of $\bar{z}$ with respect to τ are given below.

For $z/L < 0$

$$\frac{1}{\bar{z}}\frac{\partial \bar{z}}{\partial \tau} = \frac{\frac{1}{z_o}\frac{\partial z_o}{\partial \tau} + B\ \frac{1}{x}\frac{\partial x}{\partial \tau} + M\ \frac{\partial L}{L \partial \tau}}{1 + B + M}$$

with

$$B = \ln\left(0.6\ \bar{z}/z_o\right) - \psi\left(0.6\ \bar{z}/L\right)$$

$$M = \left\{\frac{1.75\ B}{1-35\bar{z}/L} - 0.6\ \left(\psi + \frac{\bar{z}}{L}\frac{\partial \psi}{\partial(\bar{z}/L)}\right)\right\}\frac{\bar{z}}{L}$$

$$\frac{\partial \psi}{\partial(\bar{z}/L)} = -45\ \left(\frac{1}{1+a} + \frac{a-1}{1+a^2}\right)\ a^{-3}$$

$$a = \left(1 - 9.0\ \bar{z}/L\right)^{1/4}$$

For $z/L \geq 0$

$$\frac{1}{\bar{z}}\ \frac{\partial \bar{z}}{\partial \tau} = \frac{\frac{1}{z_o}\frac{\partial z_o}{\partial \tau} + B\ \frac{1}{x}\ \frac{\partial x}{\partial \tau} + M\ \frac{\partial L}{L \partial \tau}}{1 + B + M}$$

with

$$B = \frac{k^2 \, x \, /\bar{z}}{0.74 \, (1 + 4.9 \, \bar{z}/L)}$$

$$M = \frac{\{49 \, (\ln \, (0.6 \, \bar{z}/z_o + 98 \, \bar{z}/L + 1) + 1.2\} \, \bar{z}/L}{1 + 4.9 \, \bar{z}/L}$$

4. Monin-Obukhov Length

The Monin-Obukhov length (L) is given by:

$$L = \begin{cases} \dfrac{-T_a u^{*3}}{g\ k\ H} & \text{for daytime} \\ A u^{*2} & \text{for nighttime } A = 1100\ s^2 \cdot m^{-1} \end{cases}$$

where

T_a is the ambient temperature (K),

g is the acceleration due to gravity ($m.s^{-2}$),

H is the surface heat flux ($K.m.s^{-1}$).

The derivatives with respect to τ are given below.

For daytime:

$$\frac{\partial L}{\partial \tau} = L \left\{ \frac{1}{T_a} \frac{\partial T_a}{\partial \tau} + \frac{3}{u^*} \frac{\partial u^*}{\partial \tau} - \frac{1}{H} \frac{\partial H}{\partial \tau} \right\}$$

For nighttime:

$$\frac{\partial L}{\partial \tau} = \frac{2L}{u^*} \frac{\partial u^*}{\partial \tau}$$

5. Friction Velocity

The friction velocity (u^*) is given by:

$$u^* = \begin{cases} \tilde{u}^* \left\{1 + \bar{f} \ln\left[1 + (1.95 + 32.6 \left(\frac{z_o}{z_m}\right)^{0.45}) \frac{H}{\tilde{H}}\right]\right\} & \text{for daytime} \\ \tilde{u}^* \left[\frac{1}{2} + \frac{1}{2} \sqrt{1 - \left(\frac{2u_o}{\sqrt{\tilde{u}^* u}}\right)^2}\right] & \text{for nighttime} \end{cases}$$

where:

u is the wind speed ($m.s^{-1}$),

z_m is the height at which the wind speed is measured (m).

$$\tilde{u}^* = \frac{ku}{\ln(z_m/z_o)}$$

$$\bar{f} = \begin{cases} 0.128 + 0.005 \ln (z_o/z_m) & \text{for } z_o/z_m \leq 0.01 \\ 0.105 & \text{for } z_o/z_m > 0.01 \end{cases}$$

$$u_o = \sqrt{\frac{\beta z_m}{kA}} \qquad \beta = 4.7,\ A = 1100$$

$$\tilde{H} = \frac{Ta\ \tilde{u}^{*3}}{g\ k\ z_m}$$

The derivatives of u^* with respect to τ are given below.

For nighttime:

$$\frac{1}{u^*} \frac{\partial u^*}{\partial \tau} = \frac{1}{\tilde{u}^*} \frac{\partial \tilde{u}^*}{\partial \tau} + \frac{\tilde{u}^*}{4} (1 - a^2)^{-1/2} a^2$$

$$\left(\frac{-1}{z_m} \frac{\partial z_m}{\partial \tau} + \frac{1}{\tilde{u}^*} \frac{\partial \tilde{u}^*}{\partial \tau} + \frac{1}{u} \frac{\partial u}{\partial \tau} \right)$$

where

$$a = \frac{2Uo}{\sqrt{\tilde{u}^* u}}$$

for daytime

$$\frac{1}{u^*}\frac{\partial u^*}{\partial \tau} = \frac{\partial \tilde{u}^*}{\tilde{u}^* \partial \tau} + \frac{\tilde{u}^*}{u^*} \ln(\chi m) \cdot \frac{\partial \bar{F}}{\partial \tau} + \frac{\tilde{u}^* \bar{F}}{u^*} \frac{\partial \ln(\chi m)}{\partial \tau}$$

where

$$\chi m = 1 + \left\{ 1.95 + 32.6 \left(\frac{Z_o}{Z_m}\right)^{0.45} \right\} \frac{H}{\tilde{H}}$$

$$\frac{1}{\tilde{u}^*}\frac{\partial \tilde{u}^*}{\partial \tau} = \frac{1}{u}\frac{\partial u}{\partial \tau} - \frac{1}{\ln(z_o/Z_m)} \cdot \frac{1}{(Z_o/Z_m)} \cdot \frac{\partial (Z_o/Z_m)}{\partial \tau}$$

$$\frac{1}{(Z_o/Z_m)}\frac{\partial (Z_o/Z_m)}{\partial \tau} = \frac{1}{Z_o}\frac{\partial Z_o}{\partial \tau} - \frac{1}{Z_m}\frac{\partial Z_m}{\partial \tau}$$

$$\frac{\partial \bar{F}}{\partial \tau} = \begin{cases} 0 & \text{for } Z_o/Z_m > 0.01 \\ 0.005 \dfrac{1}{(Z_o/Z_m)} \dfrac{\partial (Z_o/Z_m)}{\partial \tau} & \text{otherwise} \end{cases}$$

$$\frac{\partial \ln(\chi m)}{\partial \tau} = \frac{M (H/\tilde{H})}{1 + M(H/\tilde{H})} \frac{1}{(H/\tilde{H})} \frac{\partial (H/\tilde{H})}{\partial \tau} +$$

$$\frac{14.67 (Z_o/Z_m)^{0.45} (H/\tilde{H})}{1 + M(H/\tilde{H})} \cdot \frac{1}{(Z_o/Z_m)} \frac{\partial (Z_o/Z_m)}{\partial \tau}$$

$$M = 1.95 + 326 (Z_o/Z_m)^{0.45}$$

$$\frac{1}{(H/\tilde{H})}\frac{\partial (H/\tilde{H})}{\partial \tau} = \left\{ \frac{1}{H}\frac{\partial H}{\partial \tau} - \frac{1}{Ta}\frac{\partial Ta}{\partial \tau} - 3\frac{1}{\tilde{u}^*}\frac{\partial \tilde{u}}{\partial \tau} + \frac{1}{Z_m}\frac{\partial Z_m}{\partial \tau} \right\}$$

6. Surface Heat Flux

The surface heat flux (H) is given by:

$$H = \frac{(\alpha R + H_o)}{\rho C_p}$$

where

α is the surface factor

R is the solar radiation ($W.m^{-2}$)

H_o is the non-solar heating ($W.m^{-2}$)

ρ is the air density ($g.m^{-3}$)

C_p is specific heat of air ($joules.g^{-1} . K^{-1}$)

The non-solar heating (H_o) is given by:

$$H_o = -3 (8.5-CC)$$

where CC is the cloud cover (oktas)

The solar radiation (R) is given by:

$$R = 950 (\cos\phi \cos\tau \cos\ell + \sin\phi \sin\ell)$$

where

ϕ is the declination angle

τ is the angle of the sum for the time of day

ℓ is the latitude of the emission source

The declination angle (ϕ) is given by:

$$\phi = \tan^{-1} [0.4348 \sin (\pi(d-78)/180)]$$

where d is the Julian day

The hour angle (τ) is given by:

$$\tau = \pi (t-v)/1200$$

where

t is the time of day (HHMM)
v is the time of local noon (HHMM)

$$\frac{\partial H}{\partial \tau} = \frac{3}{\rho Cp} \frac{\partial cc}{\partial \tau} + \text{(Sensitivity respect time: this is discrete!)}$$

7. Sigma y (σ_y)

The crosswind standard deviations (σ_y) are calculated using Brigg's values as recommended by the National Working Group on Radionuclide Transport/2/.

$$\sigma_y = \frac{\sigma_\theta \cdot x}{(1 + \sqrt{x/2500})}$$

Where σ_θ is the standard derivation parameter which depends on the variation of wind direction.

This expression is for highly stable conditions and was chosen to give a conservative estimate of plume concentrations. The treatment of σ_θ is easily expanded to include other stability classes.

The derivative of σ_y respect to the parameter τ is given by:

$$\frac{\partial \sigma_y}{\partial \tau} = \frac{\sigma_y}{\sigma\theta} \frac{\partial \sigma_\theta}{\partial \tau} + \sigma_y \left(\frac{1}{x} + \frac{1}{\sqrt{2500x}\ (1 + \sqrt{x/2500})}\right) \frac{\partial x}{\partial \tau}$$

APPENDIX C

DESCRIPTION OF THE PROGRAMS

TABLE OF CONTENTS

	Page
1. INTRODUCTION	C-1
2. PROGRAM READTAPE	C-1
3. PROGRAM RNDOM	C-2
4. PROGRAM DCRDM	C-2
4.1 Routine RDCAS	C-3
4.2 Routine GENRAN	C-3
4.3 Routine SIMVAL	C-4
4.4 Routine CHKU	C-4
4.5 Routine CLXQ	C-4
4.6 Routine CDXQ	C-4
REFERENCES	
TABLE C.1	C-6
TABLE C.2	C-8
FIGURE	C-9
PROGRAM LISTINGS	

PROGRAM		READTAPE
PROGRAM		RNDOM
DATA		BLOCK
PROGRAM		DCRDM
SUBROUTINE	CDML	
SUBROUTINE	CDUB	
SUBROUTINE	CDUS	
SUBROUTINE	CDXQ	
SUBROUTINE	CDZB	
SUBROUTINE	CHKU	
SUBROUTINE	CLHF	
SUBROUTINE	CLJD	
SUBROUTINE	CLML	
SUBROUTINE	CLSGY	
SUBROUTINE	CLUB	
SUBROUTINE	CLUS	
SUBROUTINE	CLXQ	
SUBROUTINE	CLZB	
FUNCTION		DISC
FUNCTION		FSZB
FUNCTION		FUZB

SUBROUTINE	GENRAN
SUBROUTINE	JDAY
FUNCTION	RANDOM
SUBROUTINE	RDCAS
SUBROUTINE	SIMVAL
SUBROUTINE	SOLRAD
FUNCTION	UNIF
FUNCTION	Z12

1. INTRODUCTION

This Appendix describes the programs used to calculate the sensitivity functions of the Continuous Release Dispersion Model. The programs are written in ANSI FORTRAN. All the programs, with the exception of the READTAPE, were compiled and executed in an OLIVETTI M24 Microcomputer using MICROSOFT FORTRAN. The program READTAPE was compiled and executed in the IBM main frame. Figure C.1 gives an sketch of the structure and interrelation of the programs.

2. PROGRAM READTAPE

The program READTAPE is used to read out selected records from the meteorological tape provided by the Environment Studies and Assessment Department of Ontario Hydro. The tape contains hourly information of wind speed, wind direction, stability class, and precipitation values from January 1, 1980 to December 31, 1984. Each record has format

(1X, I3,1X,I2,1X,3I2,I3)

with structure

/_/DDD/_/HH/_/WD/WS/SC/RRR/

where

/_/ : indicates a space

DDD : is the day number of the year (1 to 366)

HH : is the hour (1 to 24)

WS : is the wind direction code (1 to 16) with the following convention: 1:N, 2:NNE, 3:NE, 4:ENE, 5:E, 6:ESE, 7:SE, 8:SSE, 9:S, 10:SSW, 11:SW, 12:WSW, 13:W, 14:WNW, 15:NW, 16:NNW

WS : is the wind speed expressed in dm/s

SC : is the stability class (1 to 6) corresponding to class A to F

RRR : is the amount of rain precipitation for the hour, expressed mm

The data form the tape was copied into the IBM main frame and separated into five files, each one containing one year of data.

The READTAPE program proceeds as follows. 1) read a record number (1 to 43848) into the program from File 8; 2) calculates the year the record belong to and reads it from File 10+NY, where NY is 1 for 1980, and 2 for 1981 and so on; 3) write the record into File 9; 4) check if there are more records in File 8; if there are go to step 1), otherwise stop. The program assumes

that the records numbers in File 8 have been ordered in ascending order.

3. PROGRAM RNDOM

To carry out the simulation study described in Section 4.2 of the report, it is desirable to randomly select the records to be read form the tape. The program RNDOM can be used for this purpose. This program generates semi-random integers uniformly distributed in an interval [a, b] based in the method given in/1/ and /2/. The program asks for the number of random integers to be generated, for the limits of the interval, and for three seeds (i.e. integers less than 30000) necessary to initialize the random number generator which could be obtained from a random number table.

The program call two functions:

- RANDOM : generates random number uniformly distributed in the interval [0,1]

- UNIF : transforms the interval [0,1] into the interval [a,b]

It is necessary to remember that the number has to be sorted before they could be input into the READTAPE program; any sort utility can be used for this purpose.

4. PROGRAM DCRDM

The purpose of the DCRDM program is to calculate the sensitivity function of the CRDM described in/3/. The variables used by the program are described in the source code list and will not be repeated here. The program first requests the name of the output file and the option to input the data; according to the selected option, it calls an appropriate routine to input or generate the meteorological data; then, the program check the value of wind speed (call subroutine CHKU), calculates the normalized concentrations (call routine CLXQ), and computes the sensitivity functions (call routine CDXQ); finally, it writes the output values (call routine WRDER).

There are three options to input or generate the meteorological data as is described below.

- To input a 'grid' of points where a maximum of three different points is allowed for each variable. The routine RDCAS is used to read the points.

- To generate the variables using predetermined frequencies distribution. In this case, the program ask for the number of points to be generated and for the seeds to initialize the random number generator. The routine GENRAN is used to generate the input values. The frequency distributions can be changed by altering the parameters of the routine GENRAN.

- To read data from the meteorological tape. The routine SIMVAL is used to read the data obtained from the tape and to generate the rest of the values necessary to calculate the normalized concentrations.

4.1 Routine RDCAS

This routine is used to read a 'grid' of the meteorological input data for the DCRMD program. It has two options.

- To input the data interactively. In this case the program ask for the number of different points for each variables (maximum of 3 per variable) and for the different values of the variable. Table C.1 gives an example of a run of the program when interactive input is desired.

- To input the data from a given input file. Table C.2 gives an example of the run of the program when the data is input from a file.

4.2 Routine GENRAN

This routine generates random values for the meteorological variables using predetermined frequency distributions. The program asks for the number of points to be generated and for three integers less than 30000 to be used as seeds for the random number generator. A uniform distribution or a fixed value is assumed for all the variables except temperature and wind speed which take seasonal values.

For a given season ISEA = 1, ..., 4, the temperature value is generated uniformly within the interval [XTA(1,ISEA), XTA(2,ISEA)]; in this way the seasonal variation of the temperature values is guaranteed. The range of temperatures can be changed by altering the vector of constants XTA.

The dependency of the wind speed values with the season is quite complicated/4/. It is not so much the range of values of wind speed that change seasonally but distribution of these values. An adequate procedure that maintain this dependency was implemented to generate the values of wind direction. The range of values were discretized in six intervals and the frequency of the intervals for season ISEA = 1, ..., 4, was obtained from /4/ and stored in the vector PWIND(6,ISEA) (these frequencies can be altered by changing the values of the matrix PWIND). For a given season ISEA, the program generates the value of wind speed by picking up one of the 6 intervals, with probabilities given in PWIND(6, ISEA), and then randomly selecting a point from this interval.

The program calls the following routines and functions:

- RANDOM: generates a random number in the interval [0,1]

- DISC: generates a random number with a given discrete distribution.

- UNIF: generates a uniform random number within a given interval.

- Z12: generates a random number which takes two possible values with given probabilities values.

4.3 Routine SIMVAL

This routine read the data obtained from the meteorological tape and generate the input values for the DCRDM program. The program randomly generates the values that are not read form the tape in the following way.

- The values of month and day are calculated from the record number using the routine JDAY.
- The values of cloud cover are obtained from the values of wind speed and stability class using the routine SOLRAD which implements the method of Pasquille/5/; the values of incoming solar radiation necessary to calculate these values are selected from an interval determined by the values of wind speed and stability class.
- The values of sigma theta are taken as the middle point of the sigma theta interval corresponding to the Pasquille Categories as given in/5/.
- The rest of the values, like down wind distance, are generated as in the GENRAN routine.

4.4 Routine CHKU

This routine is called by the DCRDM program to check the value of wind speed is a valid number i.e. that the value of U_o calculated in routine CLUS is positive.

4.5 Routine CLXQ

This routine calculates the values of the normalized concentrations. The program, together with the routines it calls, is described in/3/.

4.6 Routine CDXQ

This program calculates the sensitivity function of the normalized concentrations at the input values as is described in Appendix B. It calls the following routines.

- CDUS: calculates the derivatives of the friction velocity and heat flux with respect to the input values.
- CDML: calculates the derivatives of the Monin-Obukhov length with respect to the input values.
- CDZB: calculates the derivatives of the mean particle height with respect to the input values.
- CDUB: calculates the derivatives of the mean horizontal velocity with respect to the input values.

REFERENCES

1. Wichmann, B.A., and Hill, I.D., (1982) "Algorithm AS 183. An Efficient and Portable Pseudo-Random Number Generator", Applied Statistics V31 (188-190).

2. McLeod, A.I., (1985). "Remark ASR58. A Remark onthe Algorithm AS183", Applied Statistics V34 (198-200).

3. Wright, S.C., (1983). "Dispersion Modelling System for Exercise Pickering 83 - A User Guide". Ontario Hydro Research Division Report No. 83-278-K.

4. Tam, Y.T., (1984). "A Study of the Dispersion Climatology of the Pickering Area", Ontario Hydro Research Division Report No 84-137-K.

5. Pasquill, F., (1974). "Atmospheric Diffusion Models". John Wiley.

TABLE C.1

Example of the Program DCRDM when a 'Grid' Input is Selected

```
ENTER FILE NAME TO WRITE OUTPUT VALUES
out
DO YOU WANT TO INPUT DETERMINISTIC VALUES? (Y/N)
y
WANT TO ENTER INTERACTIVE INPUT?
y
NUMBER OF LOCATIONS: NLOC
2
ENTER COORDENATES FOR LOCATION:  1 (dwd,crs,z)
500 0 0
ENTER COORDENATES FOR LOCATION:  2 (dwd,crs,z)
2000 0 0
NUMBER OF DATES: NDATE
1
ENTER DATE NUMBER:  1 YEAR(YYYY) MONTH(MM) DAY(DD)
1985 12 22
NUMBER OF TIMES OF DAY: NTIME
1
ENTER TIME NUMBER:  1  as (HHMM)
1200
NUMBER OF CLOUD COVER VALUES: NCLOUD
1
ENTER CLOUD VALUE NUMBER:  1  (0 <= CLOUD<= 8)
4
NUMBER OF SOURCE LATITUD VALUES: NSTELAT
1
ENTER SOURCE LATITUD VALUE NUMBER:  1  (Degrees)
45
NUMBER OF SOURCE HEIGHT VALUES: NSRCHGT
1
ENTER SOURCE HEIGHT VALUE NUMBER:  1  in m
20
NUMBER OF SURFACE ROUGHNESS VALUES: NZ0
1
ENTER SURFACE ROUGHNESS VALUE NUMBER:  1  in m
.1
NUMBER OF MEASUREMENT HEIGHT VALUES: NZM
1
ENTER MEASUREMENT HEIGHT VALUE NUMBER:  1  in m
10
NUMBER OF TEMPERATURE VALUES: NTA
1
ENTER TEMPERATURE VALUE NUMBER:  1 For date 1985 12 22
0
```

NUMBER OF WIND SPEED VALUES: NWIND
1
ENTER WIND SPEED VALUE NUMBER: 1 in m/s
4
NUMBER OF SIGMA THETA: NSIGT
1
ENTER SIGMA THETA VALUE NUMBER: 1
2.1

TABLE C.2

Example of a run of the DCRDM program when a 'Grid' Input is Selected and the Data is Entered from a File

```
ENTER FILE NAME TO WRITE OUTPUT VALUES
out
DO YOU WANT TO INPUT DETERMINISTIC VALUES? (Y/N)
y
WANT TO ENTER INTERACTIVE INPUT?
n
ENTER INPUT FILE NAME
input.dat
```

INPUT FILE

```
2
500 0 0          /location
2000 0 0
1          /date
1985 12 22
1          /times
1200
1          /cloud covers
4
1          /latitud
45
1          /source height
20
1          /surface roughness
.1
1          /height values
10
1          /temperature values
0
1          /wind speed values
4
1          /sigma theta values
2.1
```

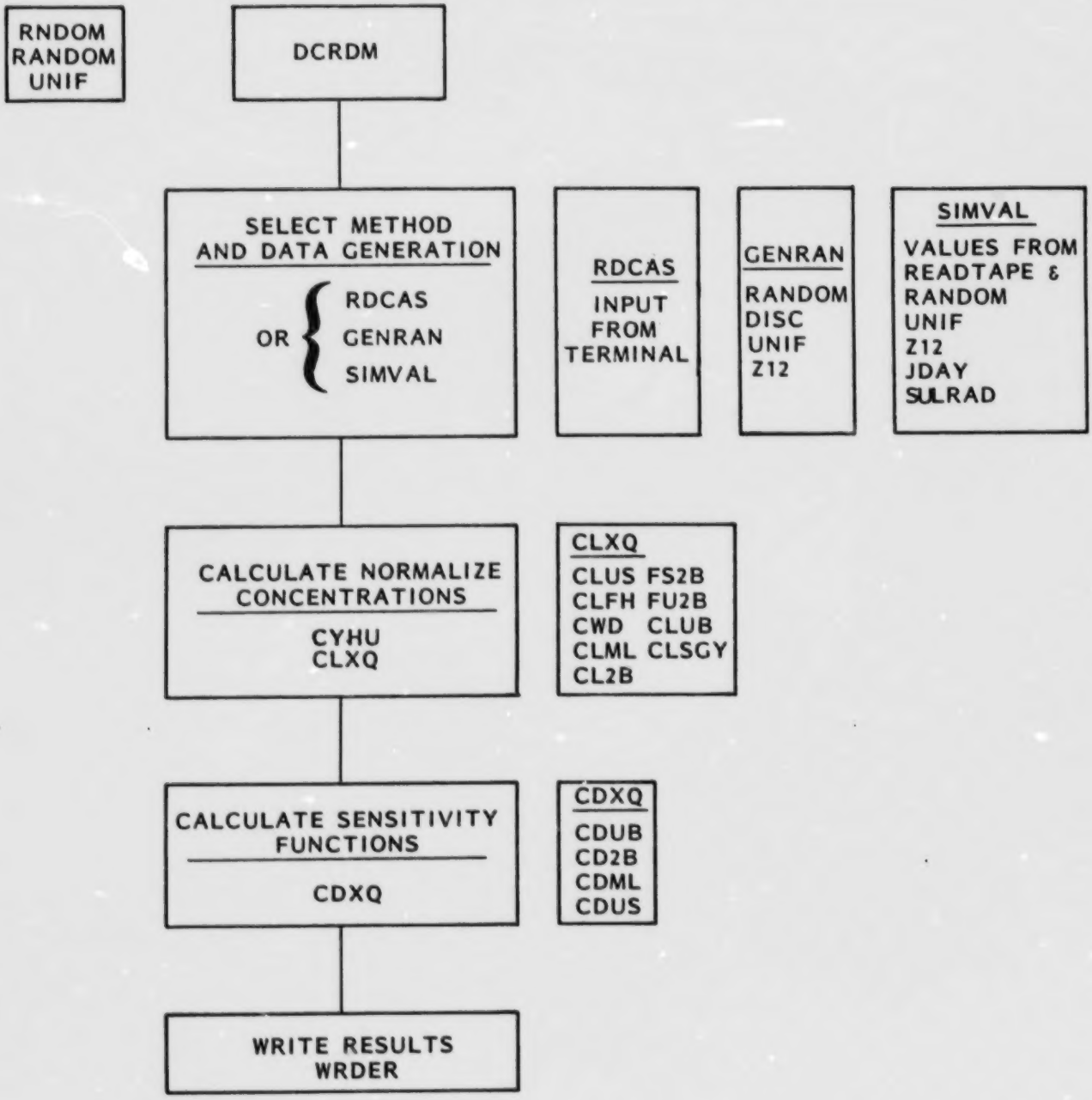

FIGURE C.1
DCRDM HIERARCHY DIAGRAM

```
 1 C-------------------------------------------------------------------
 2       PROGRAM READTAP
 3 C-------------------------------------------------------------------
 4 C  This program read the tape with the metereological data.
 5 C
 6       DIMENSION NHOUR(6), MDAT(6)
 7       CHARACTER FRMR*30, FRMW*30
 8 C
 9       DATA NHOUR/0, 8784, 15744, 26304, 35064, 43848/
10       LGUN = 8
11       LGUO = 9
12  5    FORMAT('('I4,'/,1X,I3,1X,I2,1X,3I2,I3)')
13       FRMW = '(1X,I4,1X,I3,1X,I2,1X,3I2,I3)'
14 C
15 C  Read and write data
16 C
17       NREC = 0
18       LREC = 0
19       NF = 0
20  10   READ(LGUN,*,END=1000) NRCD
21       IF(NRCD .GT. NHOUR(NF+1)) THEN
22        NF = NF + 1
23          IF(NF .GT. 5) GOTO 1000
24          NREC = NRCD - NHOUR(NF)
25          WRITE(FRMR,5) NREC
26       ELSE
27          NREC = NRCD - NHOUR(NF)
28          NDIS = NREC - LREC - 1
29          IF(NDIS .GT. 0) WRITE(FRMR, 5) NDIS
30          IF(NDIS .EQ. 0) FRMR = '(1X,I3,1X,I2,1X,3I2,I3)'
31          IF(NDIS.LT.0) GOTO 10
32       ENDIF
33       READ(NF+10, FRMR, REC=NREC) (MDAT(I),I=1,6)
34       WRITE(LGUO, FRMW) NF+1979, (MDAT(I),I=1,6)
35       LREC = NREC
36       GOTO 10
37  1000 STOP
38       END

Pass One    No Errors Detected
            38 Source Lines
```

```
      1 $DEBUG
      2 C-----------------------------------------------------------------------
      3       PROGRAM RNDOM
      4 C-----------------------------------------------------------------------
      5 C This program generates random numbers
      6 C
      7       COMMON /RAND/ IX, IY, IZ
      8       CHARACTER FNAME*20
      9       LOGICAL IFEXT
     10       LGUZ = 10
     11       WRITE(*,*) ' Enter Output file name'
     12       READ(*,10) FNAME
     13  10   FORMAT(A20)
     14       INQUIRE(FILE=FNAME,EXIST=IFEXT)
     15       IF(IFEXT) OPEN(UNIT=LGUZ,FILE=FNAME,STATUS='OLD')
     16       IF(.NOT.IFEXT) OPEN(UNIT=LGUZ,FILE=FNAME,STATUS='NEW')
     17       WRITE(*,*) ' Enter number of points to be generated'
     18       READ(*,*) N
     19       WRITE(*,*) 'Enter the limits of the rnd numbers L1 < RND < L2'
     20       READ(*,*) L1, L2
     21       FL1 = FLOAT(L1)
     22       FL2 = FLOAT(L2)
     23       WRITE(LGUZ, 15) N,L1,L2
     24  15   FORMAT(1X,I6,' RANDOM GENERATED POINTS BETWEEN '16,' AND ', I6)
     25       WRITE(*,*) ' Enter Seeds for RND : IX IY IZ (<30000)'
     26       READ(*,*) IX, IY, IZ
     27       DO 20 I =1,N
1    28          RND = RANDOM(L)
1    29          RND = UNIF(RND, FL1, FL2)
1    30          WRITE(LGUZ,*) INT(RND)
1    31   20  CONTINUE
     32       WRITE(LGUZ, 30) IX,IY,IZ
     33   30  FORMAT(' FINAL SEED (IX,IY,IZ) = ',3(I7,2X))
     34       CLOSE(LGUZ)
     35       STOP
     36       END

 Pass One    No Errors Detected
             36 Source Lines
```

```
1 $DEBUG
2 C$STORAGE:2
3 C-----------------------------------------------------------------------
4 C      BLOCK DATA
5        BLOCK DATA NDMBD
6 C-----------------------------------------------------------------------
7 C    This subroutine declares the common blocks CNSTS, STEFL, RNFL,
8 C NMSFL and METFL.
9 C
10 C CNSTS-
11 C       PI: pi (3.1415927)
12 C      RHO: dry air density at 0 deg C, 1000mb is 1276 gm/m**3
13 C       CP: specific heat at constant pressure is 1.003
14 C           joules/(gm*degK)
15 C    ALPHA: surface factor for rural area is 0.35
16 C        G: acceleration due to gravity (9.8 m/s**2)
17 C       K1: Von Karman constant is 0.35
18 C     BETA: friction velocity constant is 4.7
19 C      AUO: friction velocity and Monin-Obukhov length constant 1100.
20 C     MNTM: minimum day time (HHMM)
21 C     MXTM: maximum day time (HHMM)
22 C     LNTM: local noon time  (HHMM)
23 C
24 C STEFL-
25 C   STELAT: source latitude (degrees)
26 C   SRCHGT: source height (metres)
27 C       ZO: surface roughness (metres)
28 C       ZM: wind measurement height (metres)
29 C    MXDWN: maximum downwind distance
30 C      NUS: number of user-selected points
31 C      USP: user-selected points indicator
32 C
33 C
34 C
35 C METFL-
36 C       TA: ambient temperature (degrees K)
37 C     STAB: stability constant for sigma y (Briggs, 1973)
38 C  WIND(1): wind speed (metres)
39 C  WIND(2): wind direction (degrees from)
40 C     SIGY: sigma y values (metres)
41 C      AZB: mean particle height values (metres)
42 C      LOC: downwind, crosswind locations (metres)
43 C      XQ1: normalized concentrations (sec/m**3)
44 C     SIGT: sigma theta (radians)
45 C     RAIN: rainfall intensity ((mm/hr)
46 C     YEAR: year (yyyy)
47 C    MONTH: month (mm)
48 C      DAY: day (dd)
49 C     TIME: 24 hour time (hhmm)
50 C    CLOUD: cloud cover amount (oktas)
51 C     NLOC: number of locations
```

```
      C    LSIGT: use sigma theta indicator
      C    LRAIN: rainfall indicator
      C
            REAL PI,RHO,CP,ALPHA,G,K1,BETA,AUO
            INTEGER MNTM(12),MXTM(12),LNTM
            COMMON/CNSTS/PI,RHO,CP,ALPHA,G,K1,BETA,AUO,
           *             MNTM,MXTM,LNTM
      C
            REAL STELAT,SRCHGT,ZO,ZM,MXDWN
            INTEGER NUS
            LOGICAL USP
            COMMON/STEFL/STELAT,SRCHGT,ZO,ZM,MXDWN,NUS,USP
      C
            REAL TA,STAB,WIND(2),SIGY(10),AZB(10),LOC(10,3),XQ(10),SIGT,RAIN
            INTEGER YEAR,MONTH,DAY,TIME,CLOUD,NLOC
            LOGICAL LSIGT,LRAIN
            COMMON/METFL/TA,STAB,WIND,SIGY,AZB,LOC,XQ,SIGT,RAIN,
           *             YEAR,MONTH,DAY,TIME,CLOUD,NLOC,LSIGT,LRAIN
      C
            DATA PI/3.1415927/,RHO/1276.0/,CP/1.003/,ALPHA/0.35/,
           *     G/9.8/,K1/0.35/,BETA/4.7/,AUO/1100.0/,LNTM/1200/
      C      DATA MNTM/800,700,600,500,430,400,430,500,530,630,730,800/
      C      DATA MXTM/1600,1700,1800,1900,1930,2000,1930,1900,1830,
      C     *          1730,1630,1600/
            DATA MNTM/800,715,630,530,500,430,500,545,630,715,745,800/
            DATA MXTM/1600,1645,1715,1815,1900,1930,1900,1815,1730,
           *          1645,1615,1615/
      C
            END

Pass One    No Errors Detected
            80 Source Lines
```

```
C-----------------------------------------------------------------------
      PROGRAM DCRDM
C-----------------------------------------------------------------------
C This routine read the values of the paramter space to be used to
C calculate the sensitivity of the model. After the input values are rea
C the program calculates the  partial derivatives of the normalized conc
C
C
C This program calculates the sensitivity of the CDRM
C
C Variables definition-
C     XLOC: down wind distance
C     YLOC: crosswind distance
C     ZLOC: height at location (XLOC, YLOC)
C      DWN: downwind distance (m)
C      CRS: crosswind distance (m)
C      PSI: constant needed for UB
C      DUB: vector of normalized derivatives of ub
C      DZB: vector of normalized derivarives of zb
C      DUS: derivatives of friction velocity (m/s)
C      DML: derivatives of Monin-Obukhov length (m)
C      DXQ: derivatives of the normalized concentration
C     NPTS: number of cases to be simulated
C    LGUTI: logical terminal unit number for input
C    LGUTO: logical terminal unit number for output
C
C Output-
C     DXQ: Vector of derivatives of xq
C     The vector of the derivatives is organized as follows
C     DXQ(1): wind speed U
C     DXQ(2): Temperature TA
C     DXQ(3): Heat Flux HF
C     DXQ(5): Sourface Roughness Z0
C     DXQ(5): Measurement Height ZM
C     DXQ(6): Downwind distance
C     DXQ(7): Crosswind distance
C     DXQ(8): Height
C     DXQ(9): sigma theta
C
C External routines:
C     CHKU(LW)
C     CLSGY (X,SGY)
C     CLUS (US,HF)
C     CLHF (HF)
C     CLJD (YY,MM,DD,JD)
C     CLML (US,HF,ML)
C     CLXQ (US,UB,ML,HF)
C     CLZB (ML,CK1,ZOM,DWN,ZB)
C     CLUB (US,ML,CK1,ZOM,DWN,CRS,ZB,UB)
C     CDUS (US, HF, DUS)
C     CDML(ML, US, HF, TA, DUS,DML)
```

```
52 C     CDZB(DML, DWN, ZO, CK1, ZB, ML, DZB, PSI, DPSI)
53 C     DLUB(DUS, DML, DZB, US, ML, ZB, UB, K1, ZO, PSI, DPSI, DUB)
54 C     CDXQ (DXQ)
55 C     FUZB (ML,CK1,ZOM,DWN,X)
56 C     FSZB (ML,CK1,ZOM,DWN,X)
57 C     RDLOC(NLOC, LOC, LGUTI, LGUTO)
58 C     RDCAS (NLOC,LOC,VYEAR,VMONTH,VDAY,VTIME,VCLOUD,VSTELAT,VSRCHGT,VO
59 C    *     ,VZM,NYEAR,NMONTH,NDAY,NTIME,NCLOUD,NSTELAT,NSRCHGT,NZO,NZM
60 C    *     ,VTA,VWIND,VSIGT, NTA,NWIND,NSIGT,LGUTI, LGUTO,LGUO,FNAME)
61 C     WRDER(DXQ,IFTIT,LGUO)
62 C     GENRAN(STELAT,SRCHGT,ZO,ZM,TA,STAB,WIND,XLOC,
63 C    *       SIGT,YEAR,MONTH,DAY,TIME,CLOUD)
64 C     SIMVAL(STELAT,SRCHGT,ZO,ZM,TA,STAB,WIND,XLOC,
65 C    *    SIGT,YEAR,MONTH,DAY,TIME,CLOUD,LNTM,STELAT,PI,LGUF,IFEND)
66 C     FUNCTION DISC(RND, N, XV, FX)
67 C     FUNCTION UNIF(RND, A, B)
68 C     FUNCTION Z12(RND, X1, X2, P)
69 C     FUNCTION RANDOM(L)
70 C
71 C   Fernando Camacho
72 C   Statistics and Reliability Section (KR297)
73 C   Operation Research Department
74 C   Ontario Hydro Research Division
75 C   800 Kipling Ave. Toronto ONT
76 C   M8Z 5S4
77 C   Phone: (416)-211-4111 Ext 6078
78 C
79       REAL DUS(5), DML(5), DZB(6), DUB(6), DXQ(9,10)
80       LOGICAL IFTIT, IFEXT, NFXLC
81       CHARACTER FNAME*80, ANSWER
82       REAL ML, US, UB(10), HF, HS
83       COMMON /RESULTS/ ML, US, UB, HF, HS
84       COMMON /RAND/ IX, IY, IZ
85 C
86       REAL VTA(3,3),VWIND(3),VSIGT(3)
87       INTEGER VYEAR(3),VMONTH(3),VDAY(3),VTIME(3),VCLOUD(3)
88       REAL VSTELAT(3),VSRCHGT(3),VZO(3),VZM(3)
89 C
90       REAL PI,RHO,CP,ALPHA,G,K1,BETA,AUO
91       INTEGER MNTM(12),MXTM(12),LNTM
92       COMMON/CNSTS/PI,RHO,CP,ALPHA,G,K1,BETA,AUO,
93      *             MNTM,MXTM,LNTM
94 C
95       REAL STELAT,SRCHGT,ZO,ZM,MXDWN
96       INTEGER NUS
97       LOGICAL USP
98       COMMON/STEFL/STELAT,SRCHGT,ZO,ZM,MXDWN,NUS,USP
99 C
100      REAL TA,STAB,WIND(2),SIGY(10),AZB(10),LOC(10,3),XQ(10),SIGT,RAIN
101      INTEGER YEAR,MONTH,DAY,TIME,CLOUD,NLOC
102      LOGICAL LSIGT,LRAIN
```

```
      COMMON/METFL/TA,STAB,WIND,SIGY,AZB,LOC,XQ,SIGT,RAIN,
     *             YEAR,MONTH,DAY,TIME,CLOUD,NLOC,LSIGT,LRAIN
C
      EQUIVALENCE (NDATE,NYEAR)
      LGUTI = 0
      LGUTO = 0
      LGUO = 5
      LGUF = 8
      fname = 'dump'
      INQUIRE(FILE=FNAME,EXIST=IFEXT)
      IF(IFEXT) OPEN(UNIT=10,FILE=FNAME,STATUS='OLD')
      IF(.NOT.IFEXT) OPEN(UNIT=10,FILE=FNAME,STATUS='NEW')
C
C Read the parameters values
C
      STAB = 0.0
      NLOC = 1
      LSIGT = .TRUE.
      LRAIN = .FALSE.
      IFTIT = .TRUE.
C
C Read output file name
C
      WRITE(LGUTO, *) ' ENTER FILE NAME TO WRITE OUTPUT VALUES'
      READ(LGUTI, 10) FNAME
 10   FORMAT(A)
      INQUIRE(FILE=FNAME,EXIST=IFEXT)
      IF(IFEXT) OPEN(UNIT=LGUO,FILE=FNAME,STATUS='OLD')
      IF(.NOT.IFEXT) OPEN(UNIT=LGUO,FILE=FNAME,STATUS='NEW')
C
C  Decide if read observations or generate them randomly
C
      WRITE(LGUTO, *)
     1      ' DO YOU WANT TO INPUT DETERMINISTIC VALUES? (Y/N)'
      READ(LGUTI, 10) ANSWER
      IF (ANSWER .EQ. 'y' .OR. ANSWER .EQ. 'Y') GOTO 50
      WRITE(LGUTO, *)
     1      ' DO YOU WANT TO GENERATE RANDOM VALUES? (Y/N)'
      READ(LGUTI, 10) ANSWER
      IF (ANSWER .EQ. 'y' .OR. ANSWER .EQ. 'Y') GOTO 100
      IFTAPE = 1
      WRITE(LGUTO, *)
     1  ' DO YOU WANT TO INPUT RANDOM VALUES OBTAINED FROM TAPE? (Y/N)'
      READ(LGUTI, 10) ANSWER
      IF (ANSWER .EQ. 'y' .OR. ANSWER .EQ. 'Y') GOTO 150
      STOP
C
C Proceed reading deterministic parameter values
C
 50   CONTINUE
      CALL RDCAS(NLOC,LOC,VYEAR,VMONTH,VDAY,VTIME,VCLOUD,VSTELAT,VSRCHGT
```

```
     154      *,VZO,VZM,NYEAR,NMONTH,NDAY,NTIME,NCLOUD,NSTELAT,NSRCHGT,NZO,NZM
     155      *,VTA,VWIND,VSIGT, NTA,NWIND,NSIGT,LGUTI,LGUTO,LGUO,FNAME)
     156 C
     157       DO 30 IZO = 1, NZO
   1 158       DO 30 IZM = 1, NZM
   2 159       DO 30 IWIND = 1, NWIND
   3 160          ZO = VZO(IZO)
   3 161          ZM = VZM(IZM)
   3 162          WIND(1) = VWIND(IWIND)
   3 163          CALL CHKU(LGUTO)
   3 164       DO 20 IDATE = 1,NDATE
   4 165       DO 20 ITIME = 1,NTIME
   5 166       DO 20 ICLOUD = 1, NCLOUD
   6 167       DO 20 ISTELAT = 1, NSTELAT
   7 168       DO 20 ISRCHGT = 1, NSRCHGT
   8 169       DO 20 ITA = 1, NTA
   9 170       DO 20 ISIGT = 1, NSIGT
   * 171          YEAR = VYEAR(IDATE)
   * 172          MONTH = VMONTH(IDATE)
   * 173          DAY = VDAY(IDATE)
   * 174          TIME = VTIME(ITIME)
   * 175          CLOUD = VCLOUD(ICLOUD)
   * 176          STELAT = VSTELAT(ISTELAT)
   * 177          SRCHGT = VSRCHGT(ISRCHGT)
   * 178          TA = VTA(IDATE,ITA)
   * 179          SIGT = VSIGT(ISIGT)
   * 180 C
   * 181 C Calculate the normalized values
   * 182 C
   * 183 C        write(*,*)  ' Call clxq'
   * 184          CALL CLXQ
   * 185 C
   * 186 C Calculate the partial derivatives of the normalized constant
   * 187 C
   * 188 C        write(*,*)  ' Call CDXQ '
   * 189          CALL CDXQ(DXQ)
   * 190 C
   * 191 C Write out the partial derivatives
   * 192 C
   * 193          CALL WRDER(DXQ,IFTIT,LGUO)
   * 194 C
   * 195  20   CONTINUE
   3 196  30   CONTINUE
     197       GOTO 200
     198 C
     199 C Proceed generating the parameter values
     200 C
     201  100  CONTINUE
     202       WRITE(LGUTO, *)  ' ENTER NUMBER OF POINTS TO BE GENERATED'
     203       READ(LGUTI, *) NPTS
     204       WRITE(LGUTO, *)  ' Enter the SEEDS of the RND number generator'
```

```
      WRITE(LGUTO, *)  ' Enter IX  IY  IZ    ( < 30000 )'
      READ(LGUTI, *)  IX, IY, IZ
      ZM = 10.0
      STELAT = 45.0
      YEAR = 1985
      LOC(1,2) = 0.0
      LOC(1,3) = 0.0
      DO 110 ICAS = 1, NPTS
1     c     write(10,*) ' Icas IX IY IZ', ICAS, IX, IY, IZ
1           CALL GENRAN(STELAT,SRCHGT,Z0,ZM,TA,STAB,XWIND,XLOC,
1     *                 SIGT,YEAR,MONTH,DAY,TIME,CLOUD)
1           LOC(1,1) = XLOC
1           WIND(1) = XWIND
1     c      write(*,*) ' z0 zm loc wind ta' ,z0,zm,xloc,wind(1),ta
1           CALL CHKU(LGUTO)
1     C
1     C Calculate the normalized values
1     C
1           CALL CLXQ
1     C
1     C Calculate the partial derivatives of the normalized constant
1     C
1           CALL CDXQ(DXQ)
1     C
1     C Write out the partial derivatives
1     C
1           CALL WRDER(DXQ,IFTIT,LGUO)
1     C
1      110  CONTINUE
      C
      C Proceed reading data generated from the metereological TAPE
      C
       150  CONTINUE
            WRITE(LGUTO, *)  ' ENTER DATA FILENAME'
       160  READ(LGUTI, 10) FNAME
            INQUIRE(FILE=FNAME,EXIST=IFEXT)
            IF(IFEXT) THEN
              OPEN(UNIT=LGUF,FILE=FNAME)
            ELSE
              WRITE(LGUTO, *) ' FILE DOES NOT EXIST. PLEASE ENTER FILENAME'
              GOTO 160
            ENDIF
            WRITE(LGUTO, *)  ' Enter the SEEDS of the RND number generator'
            WRITE(LGUTO, *)  ' Enter IX  IY  IZ    ( < 30000 )'
            READ(LGUTI, *)  IX, IY, IZ
            WRITE(LGUTO, *) ' Do you want a fix location?'
            READ(LGUTI, 10) ANSWER
            NFXLC = .TRUE.
            IF (ANSWER .EQ. 'y' .OR. ANSWER .EQ. 'Y') NFXLC = .FALSE.
            IF(NFXLC) GOTO 165
              WRITE(LGUTO, *) ' Enter value of XLOC'
```

```
256         READ(LGUTI, *) XFXLC
257         LOC(1,1) = XFXLC
258  165  CONTINUE
259       ZM = 10.0
260       STELAT = 45.0
261       LOC(1,2) = 0.0
262       LOC(1,3) = 0.0
263 C
264 C  Read values and generate input
265 C
266       ICAS = 0
267       IFEND = 0
268  170  CALL SIMVAL(STELAT,SRCHGT,ZO,ZM,TA,STAB,XWIND,XLOC,SIGT,YEAR,
269      *            MONTH,DAY,TIME,CLOUD,LGUF,IFEND)
270       IF(IFEND .NE. 0) GOTO 200
271       ICAS = ICAS + 1
272 C
273 C  Obtain the derivatives
274 C
275 c      write(10,*) ' Icas IX IY IZ', ICAS, IX, IY, IZ
276       IF(NFXLC) LOC(1,1) = XLOC
277       WIND(1) = XWIND
278 c     write(10,*) ' loc wind ta cloud' ,xloc,wind(1),ta,cloud
279       CALL CHKU(LGUTO)
280 C
281 C Calculate the normalized values
282 C
283       CALL CLXQ
284 C
285 C Calculate the partial derivatives of the normalized constant
286 C
287       CALL CDXQ(DXQ)
288 C
289 C Write out the partial derivatives
290 C
291       CALL WRDER(DXQ,IFTIT,LGUO)
292 C
293       GOTO 170
294  200  CONTINUE
295       close(unit=10)
296       CLOSE(UNIT=LGUO)
297       IF(TAPE .EQ. 1) CLOSE(UNIT=LGUF)
298       STOP
299       END

Pass One    No Errors Detected
          299 Source Lines
```

```
C-----------------------------------------------------------------------
      SUBROUTINE CDML(ML, US, HF, DUS,DML)
C-----------------------------------------------------------------------
C    This subroutine calculates the normalized derivatives of
C    the Monin-Obukhov length.
C
C Input-
C     DUS: vector of derivatives of the friction velocity (m/s)
C      ML: Monin-Obukhov length
C      US: friction velocity
C      TA: temperature
C      HF: heat flux
C Output-
C     DML: vector of derivatives of the Monin-Obukhov length (m)
C        = D(ML)/ML
C
      REAL ML, DUS(6), DML(6)
C
      REAL PI,RHO,CP,ALPHA,G,K1,BETA,AUO
      INTEGER MNTM(12),MXTM(12),LNTM
      COMMON/CNSTS/PI,RHO,CP,ALPHA,G,K1,BETA,AUO,
     *             MNTM,MXTM,LNTM
C
      REAL TA,STAB,WIND(2),SIGY(10),AZB(10),LOC(10,3),XQ(10),SIGT,RAIN
      INTEGER YEAR,MONTH,DAY,TIME,CLOUD,NLOC
      LOGICAL LSIGT,LRAIN
      COMMON/METFL/TA,STAB,WIND,SIGY,AZB,LOC,XQ,SIGT,RAIN,
     *             YEAR,MONTH,DAY,TIME,CLOUD,NLOC,LSIGT,LRAIN
C
      IF((TIME.LT.MNTM(MONTH)).OR.(TIME.GT.MXTM(MONTH)).OR.
     *    (STAB.NE.0.AND.STAB.LE.0.06)) GOTO 10
C
C Calculate the Monin-Obukhov length for unstable conditions
C
      DML(1) = 3.0*DUS(1)
      DML(2) = 3.0*DUS(2) + 1.0/TA
      DML(3) = 3.0*DUS(3) - 1.0/HF
      DML(4) = 3.0*DUS(4)
      DML(5) = 3.0*DUS(5)
      DML(6) = 0.0
      RETURN
C
C Calculate the Monin-Obukhov length for nighttime
C
 10   CONTINUE
      DML(1) = 2.0*DUS(1)
      DML(2) = 2.0*DUS(2)
      DML(3) = 2.0*DUS(3)
      DML(4) = 2.0*DUS(4)
      DML(5) = 2.0*DUS(5)
      DML(6) = 0.0
```

```
   52 C
   53       RETURN
   54       END

Pass One    No Errors Detected
            54 Source Lines
```

```
      1 C-----------------------------------------------------------------------
      2       SUBROUTINE CDUB(DUS, DML, DZB, US, ML, ZB, UB, K1,
      3     *                 Z0, PSI, DPSI, DUB)
      4 C-----------------------------------------------------------------------
      5 C    This subroutine calculates the derivatives of mean horizontal veloc
      6 C
      7 C Input-
      8 C     DUS: derivatives of friction velocity (m/s)
      9 C     DML: derivatives of Monin-Obukhov length (m)
     10 C     DZB: derivatives of mean particle height
     11 C      US: friction velocity (m/s)
     12 C      ML: Monin-Obukhov length (m)
     13 C      ZB: mean particle height
     14 C      UB: mean horizontal velocity
     15 C      K1: Von Karman constant
     16 C      Z0: surface roughness (m)
     17 C     PSI: constant needed for UB
     18 C    DPSI: derivative of psi respect to zb/l
     19 C
     20 C Output-
     21 C     DUB: Vector of derivatives of mean horizontal velocity (m/s)
     22 C
     23 C
     24       REAL DUS(5), DML(6), DZB(6), DUB(6), DZL(6)
     25       REAL US,ML,ZB,UB,K1,Z0,PSI,DPSI
     26 C
     27       ZL = ZB/ML
     28       DO 10 I=1,6
  1  29          DZL(I) = (DZB(I) - DML(I))*ZL
  1  30  10   CONTINUE
     31 C
     32 C  Calculate the derivatives for zl/ml < 0
     33 C
     34       C1 = US/(K1*UB)
     35       C2 = 4.7/UB
     36       IF(ZL .LT. 0.0) C2 = -0.6*(PSI + DPSI*ZL)*C1
     37          DO 20 I = 1,5
  1  38             DUB(I) = DUS(I) + C1*DZB(I) + C2*DZL(I)
  1  39  20      CONTINUE
     40          DUB(6) = C1*DZB(6) + C2*DZL(6)
     41       RETURN
     42       END

 Pass One    No Errors Detected
             42 Source Lines
```

```
C-----------------------------------------------------------------------
      SUBROUTINE CDUS (US, HF, DUS)
C-----------------------------------------------------------------------
C     This routine calculates the deriative of us
C
C Input -
C      US: friction velocity
C      HF: heat flux value
C
C Output-
C    DUS( ): Vector of the normalized derivative of friction velocity
C         = D(US)/US where D(.) denotes the partial derivative
C
      REAL US, HF, DZOZM(5), DUSS(5), DUS(5), DLGN(5), DHHS(5)
C
      REAL PI,RHO,CP,ALPHA,G,K1,BETA,AUO
      INTEGER MNTM(12),MXTM(12),LNTM
      COMMON/CNSTS/PI,RHO,CP,ALPHA,G,K1,BETA,AUO,
     *             MNTM,MXTM,LNTM
C
      REAL STELAT,SRCHGT,ZO,ZM,MXDWN
      LOGICAL USP
      COMMON/STEFL/STELAT,SRCHGT,ZO,ZM,MXDWN,MUS,USP
C
      REAL TA,STAB,WIND(2),SIGY(10),AZB(10),LOC(10,3),XQ(10),SIGT,RAIN
      INTEGER YEAR,MONTH,DAY,TIME,CLOUD,NLOC
      LOGICAL LSIGT,LRAIN
      COMMON/METFL/TA,STAB,WIND,SIGY,AZB,LOC,XQ,SIGT,RAIN,
     *             YEAR,MONTH,DAY,TIME,CLOUD,NLOC,LSIGT,LRAIN
C
C
C Calculate the DZOZM = D(ZO/ZM)/(ZO/ZM)
C
        ZOZM = ZO/ZM
C       DZOZM(1) = 0.
C       DZOZM(2) = 0.
C       DZOZM(3) = 0.
        DZOZM(4) = 1.0/ZO
        DZOZM(5) = -1.0/ZM
C
C Calculate DUSS = D(USS)/USS
C
        USS = -K1*WIND(1)/ALOG(ZOZM)
        DUSS(1) = 1.0/WIND(1)
C       DUSS(2) = 0.
C       DUSS(3) = 0.
        DUSS(4) = -DZOZM(4)/ALOG(ZOZM)
        DUSS(5) = -DZOZM(5)/ALOG(ZOZM)
C
C Calculate the derivatives of US for daytime
C
```

```
      IF((TIME.LT.MNTM(MONTH)).OR.(TIME.GT.MXTM(MONTH)).OR.
     *       (STAB.NE.0.AND.STAB.LE.0.06)) GOTO 10
C
      IF (ZOZM .LE. 0.01) THEN
         FB = 0.128 + (0.005*ALOG(ZOZM))
         DFB4 = 0.005*DZOZM(4)
         DFB5 = 0.005*DZOZM(5)
      ELSE
         FB = 0.105
         DFB4 = 0.0
         DFB5 = 0.0
      ENDIF
C
      HS = TA*USS*3.0/(K1*G*ZM)
      DHHS(1) = -3.0*DUSS(1)
      DHHS(2) = -1.0/TA
      DHHS(3) = 1.0/HF
      DHHS(4) = -3.0*DUSS(4)
      DHHS(5) = (1.0/ZM) - 3.0*DUSS(5)
C
      ZOZM5 = ZOZM**0.45
      HH = HF/HS
      XM = 1.95 + 32.6*ZOZM5
      XMM = 1.0 + (XM*HH)
      XM2 = 14.67*ZOZM5*HH
      C1 = XM*HH/XMM
      C2 = XM2/XMM
      DLGN(1) = C1*DHHS(1)
      DLGN(2) = C1*DHHS(2)
      DLGN(3) = C1*DHHS(3)
      DLGN(4) = (C1*DHHS(4))+(C2*DZOZM(4))
      DLGN(5) = (C1*DHHS(5))+(C2*DZOZM(5))
C
      UFB = USS*FB/US
      ULN = USS*ALOG(XMM)/US
      DUS(1) = (UFB*DLGN(1)) + DUSS(1)
      DUS(2) = UFB*DLGN(2)
      DUS(3) = UFB*DLGN(3)
      DUS(4) = (UFB*DLGN(4)) + (ULN*DFB4) + DUSS(4)
      DUS(5) = (UFB*DLGN(5)) + (ULN*DFB5) + DUSS(5)
C
      RETURN
C
C CALCULATE THE DERIVATIVE FOR NIGHT TIME
C
 10   CONTINUE
      U0 = SQRT(BETA*ZM/(AU0*K1))
      A = ((2*U0)**2)/(USS*WIND(1))
      IF ((1.-A) .GT. 0.0) THEN
         A = USS*A/(US*SQRT(1. - A)*4.0)
      ELSE
```

```
103             A = 1.0E+20
104           ENDIF
105           OPA = 1.0 + A
106           DUS(1) = (OPA*DUSS(1)) + (A/WIND(1))
107           DUS(2) = 0.0
108           DUS(3) = 0.0
109           DUS(4) = OPA*DUSS(4)
110           DUS(5) = (OPA*DUSS(5)) - (A/ZM)
111           RETURN
112           END

Pass One    No Errors Detected
          112 Source Lines
```

```
C-----------------------------------------------------------------------
      SUBROUTINE CDXQ(DXQ)
C-----------------------------------------------------------------------
C    This subroutine calculates the derivatives of the concentrations.
C
C  Working vectors
C    DUS: vector of derivatives of u*
C    DML: vector of derivatives of ml
C    DUB: vector of normalized derivatives of ub (depend on loc)
C    DZB: vector of normalized derivarives of zb (depend on loc)
C    SGY: value of sigma correspondent to downwind distance x
C    DWN: downwind distance
C    CRS: crosswind distance
C      Z: height at location (DWN,CRS)
C     XQ: normalized concentration value at (DWN,CRS)
C     ZB: value of zb at (DWN,CRS)
C
C Output-
C     DXQ: Vector of derivatives of xq
C
C Externals-
C       CDUS (US, HF, DUS)
C       CDML(ML, US, HF, DUS,DML)
C       CDZB(DML, DWN, ZO, K1, ZB, ML, DZB, PSI, DPSI)
C       CDUB(DUS, DML, DZB, US, ML, ZB, UB1, K1, ZO, PSI, DPSI,
C
      REAL DUS(5), DML(6), DZB(6), DUB(6), DXQ(9,10)
      REAL ML, US, UB(10), HF, HS
      COMMON /RESULTS/ ML, US, UB, HF, HS

C
      REAL PI,RHO,CP,ALPHA,G,K1,BETA,AUO
      INTEGER MNTM(12),MXTM(12),LNTM
      COMMON/CNSTS/PI,RHO,CP,ALPHA,G,K1,BETA,AUO,
     *             MNTM,MXTM,LNTM
C
      REAL STELAT,SRCHGT,ZO,ZM,MXDWN
      INTEGER NUS
      LOGICAL USP
      COMMON/STEFL/STELAT,SRCHGT,ZO,ZM,MXDWN,NUS,USP
C
      REAL TA,STAB,WIND(2),SIGY(10),AZB(10),LOC(10,3),XQ(10),SIGT,RAIN
      INTEGER YEAR,MONTH,DAY,TIME,CLOUD,NLOC
      LOGICAL LSIGT,LRAIN
      COMMON/METFL/TA,STAB,WIND,SIGY,AZB,LOC,XQ,SIGT,RAIN,
     *             YEAR,MONTH,DAY,TIME,CLOUD,NLOC,LSIGT,LRAIN
C
      EQUIVALENCE (YLOC, CRS), (XLOC, DWN)
C
C Calculate the partial derivatives of US
C
```

```
      C      write(*,*) ' enter routine DXQ'
            CALL CDUS (US, HF, DUS)
      C
      C Calculate the partial derivatives of ML
      C
            CALL CDML(ML, US, HF, DUS,DML)
      C
      C For each location calculate the derivative of ZB, UB and XQ
      C
            DO 20 ILOC = 1,NLOC
     1      IF(XQ(ILOC) .EQ. 0.0) THEN
     1         DO 5 I=1,9
     2            DXQ(I,ILOC) = 0.0
     2    5    CONTINUE
     1      ELSE
     1      C      write(*,*) ' calculate the derivative of ZB'
     1         ZB = AZB(ILOC)
     1         UB1 = UB(ILOC)
     1         XQ1 = XQ(ILOC)
     1         DWN = LOC(ILOC,1)
     1         CRS = LOC(ILOC,2)
     1         ZLOC = LOC(ILOC,3)
     1         XSIGY = SIGY(ILOC)
     1      C
     1         CALL CDZB(DML, DWN, Z0, K1, ZB, ML, DZB, PSI, DPSI)
     1         CALL CDUB(DUS, DML, DZB, US, ML, ZB, UB1, K1,
     1     *             Z0, PSI, DPSI, DUB)
     1      C
     1      C Calculate the partial derivatives of the normalized concentrations
     1      C
     1         C1 = 0.0
     1         IF(ZLOC.NE.0.0) C1 = 1.5*(0.66*ZLOC/ZB)**1.5
     1         C2 = (1. - C1)
     1         C3 = (1. - (YLOC/XSIGY)**2)
     1         DO 10 I=1,6
     2            DXQ(I,ILOC) = -XQ1*(DUB(I) + C2*DZB(I))
     2   10    CONTINUE
     1      C
     1      C Calculate the derivatives that depend on sigma y
     1      C
     1         DY6 = 1./DWN - 1./(2.0*SQRT(2500.*DWN)*(1. + SQRT(DWN/2500)))
     1         DY9 = 1./SIGT
     1      C
     1         DXQ(6,ILOC) = DXQ(6,ILOC) - XQ1*C3*DY6
     1         DXQ(7,ILOC) = 0.0
     1         IF(YLOC .NE. 0.0) DXQ(7,ILOC) = -XQ1*(1.0 - C3)/YLOC
     1         DXQ(8,ILOC) = 0.0
     1         IF(ZLOC .NE. 0.0) DXQ(8,ILOC) = -XQ1*C1/ZLOC
     1         DXQ(9,ILOC) = -XQ1*C3*DY9
     1      ENDIF
     1   20 CONTINUE
```

```
103         RETURN
104         END

Pass One    No Errors Detected
            104 Source Lines
```

```
C-----------------------------------------------------------------------
      SUBROUTINE CDZB(DML, DWN, ZO, CK1, ZB, ML, DZB, PSI, DPSI)
C-----------------------------------------------------------------------
C    This subroutine calculates the mean particle height
C
C Input-
C     DML: Vector of the derivatives of the Monin-Obukhov length (m)
C     DWN: down wind distance (m)
C     CK1: Von Karman constant
C     ZO: surface roughness (m)
C     ZB: estimated value of zb mean particle height
C     ML: Monin-Obukhov length
C
C Output-
C     DZB: vector of derivatives of the mean particle height
C        = D(ZB)/ZB
C     PSI: constant needed for UB
C     DPSI: derivative of psi respect to zb/ml
C
      REAL ML, DZB(6), DML(6)
C
      ZL = ZB/ML
C
      IF(ZL .LT. 0) THEN
C
C     Calculate the coefficients for zb/ml < 0
C
          A = (1. - 9.0*ZL)**0.25
          PSI = 2.*ALOG((1.+A)/2.) + ALOG((1.+A**2)/2.) -
     *          2.*ATAN(A) + 3.1415927/2
          DPSI = -4.5*(1./(1.+A) + (A-1.)/(1.+A**2))/A**3
C
c      write(*,*) 'zb in dzb with zl < 0 = ',zb
          XB = ALOG(0.6*ZB/ZO) - (0.6*PSI*ZL)
          XM = (1.75*XB/(1.-3.5*ZL) - 0.6*(PSI+ZL*DPSI))*ZL
      ELSE
C
C     Calculate the coefficients for zb/ml >= 0
C
c      write(*,*) 'zb in dzb with zl >= 0 = ',zb
          XB = (CK1**2)*DWN/(ZB*0.74*(1.+4.9*ZL))
          XM = (4.9*(ALOG(0.6*ZB/ZO)+9.8*ZL+1.)+1.2)*ZL/(1.+4.9*ZL)
      ENDIF
C
      XMB = 1.0 + XB + XM
      CM = XM/XMB
C
      DZB(1) = CM*DML(1)
      DZB(2) = CM*DML(2)
      DZB(3) = CM*DML(3)
      DZB(4) = CM*DML(4) + 1./(XMB*ZO)
```

```
52        DZB(5) = CM*DML(5)
53        DZB(6) = XB/(DWN*XMB)
54        RETURN
55        END


Pass One    No Errors Detected
            55 Source Lines
```

```
C-----------------------------------------------------------------------
      SUBROUTINE CHKU(LW)
C-----------------------------------------------------------------------
C
C     This subroutine checks for the minimum u.  If wind speed is
C less than minimum u, windspeed is equal to minimum u
C
      INTEGER LW
C
      REAL PI,RHO,CP,ALPHA,G,K1,BETA,AUO
      INTEGER MNTM(12),MXTM(12),LNTM
      COMMON/CNSTS/PI,RHO,CP,ALPHA,G,K1,BETA,AUO,
     *             MNTM,MXTM,LNTM
C
      REAL STELAT,SRCHGT,ZO,ZM,MXDWN
      INTEGER NUS
      LOGICAL USP
      COMMON/STEFL/STELAT,SRCHGT,ZO,ZM,MXDWN,NUS,USP
C
      REAL TA,STAB,WIND(2),SIGY(10),AZB(10),LOC(10,3),XQ(10),SIGT,RAIN
      INTEGER YEAR,MONTH,DAY,TIME,CLOUD,NLOC
      LOGICAL LSIGT,LRAIN
      COMMON/METFL/TA,STAB,WIND,SIGY,AZB,LOC,XQ,SIGT,RAIN,
     *             YEAR,MONTH,DAY,TIME,CLOUD,NLOC,LSIGT,LRAIN
C
      UO=SQRT(BETA*ZM/(K1*AUO))
      CDN=K1/ALOG(ZM/ZO)
      UMIN=2*UO/SQRT(CDN)
C
C     IF(WIND(1).LT.UMIN)THEN
      IF(WIND(1).LT.UMIN)  WIND(1)=UMIN
C        CALL MVCSR(LW,22,70,
C    *'                                                              ',1)
C        WRITE(LW,20) WIND(1),UMIN
C20      FORMAT('   Windspeed ',F6.1,' is lower than minimum value',
C    *          ', ',F6.1,' is used'\)
C        WIND(1)=UMIN
C     ELSE
C         CALL MVCSR(LW,22,70,
C    *'                                                              ',1)
C         WRITE(LW,25)
C25       FORMAT(70(' ')\)
C     END IF
C
C
      RETURN
      END

Pass One    No Errors Detected
            47 Source Lines
```

```
C-----------------------------------------------------------------------
      SUBROUTINE CLHF(HF)
C-----------------------------------------------------------------------
C    This subroutine calculates the heat flux.
C
C Output-
C     HF: heat flux (degK*m/s)
C
C Externals-
C     CLJD(YY,MM,DD,JD)
C
      REAL HF
C
      REAL PI,RHO,CP,ALPHA,G,K1,BETA,AUO
      INTEGER MNTM(12),MXTM(12),LNTM
      COMMON/CNSTS/PI,RHO,CP,ALPHA,G,K1,BETA,AUO,
     *             MNTM,MXTM,LNTM
C
      REAL STELAT,SRCHGT,ZO,ZM,MXDWN
      INTEGER NUS
      LOGICAL USP
      COMMON/STEFL/STELAT,SRCHGT,ZO,ZM,MXDWN,NUS,USP
C
      REAL TA,STAB,WIND(2),SIGY(10),AZB(10),LOC(10,3),XQ(10),SIGT,RAIN
      INTEGER YEAR,MONTH,DAY,TIME,CLOUD,NLOC
      LOGICAL LSIGT,LRAIN
      COMMON/METFL/TA,STAB,WIND,SIGY,AZB,LOC,XQ,SIGT,RAIN,
     *             YEAR,MONTH,DAY,TIME,CLOUD,NLOC,LSIGT,LRAIN
C
      REAL RF(9)
      DATA RF/1.0,0.89,0.81,0.76,0.72,0.67,0.59,0.45,0.23/
C
C Calculate the Julian day
C
      CALL CLJD(YEAR,MONTH,DAY,JD)
C
      PHI=ATAN(0.4348*SIN(PI*(FLOAT(JD)-78.0)/180.0))
      TAU=PI*(TIME-LNTM)/1200.
C
      RLAT=STELAT*PI/180.
C
C Calculate the solar radiation (Watts or joules/s)
C
      IC=CLOUD+1
      R=RF(IC)*950*(COS(PHI)*COS(TAU)*COS(RLAT)+SIN(PHI)*SIN(RLAT))
      HO=-3*(8.5-CLOUD)
c       write(*,*) ' from clxf: R, HO, PHI, TAU, RLAT ALPHA,RHO,CP',
c     *    R, HO, PHI, TAU, RLAT, ALPHA,RHO,CP
C Calculate the heat flux
C
      HF=(ALPHA*R+HO)/(RHO*CP)
```

```
 52 c      write(*,*) ' From CLHF: CLOUD,STELAT,YEAR,MONTH,DAY TIME hf= ',
 53 C     * CLOUD,STELAT,YEAR,MONTH,DAY,TIME,HF
 54 C
 55        RETURN
 56        END

Pass One    No Errors Detected
            56 Source Lines
```

```
      1 C-----------------------------------------------------------------------
      2       SUBROUTINE CLJD(YY,MM,DD,JD)
      3 C-----------------------------------------------------------------------
      4 C    This subroutine calculates the Julian day.
      5 C
      6 C    Input-
      7 C       YY: year
      8 C       MM: month
      9 C       DD: day
     10 C
     11 C   Output-
     12 C       JD: Julian day
     13 C
     14       INTEGER YY,MM,DD,JD
     15 C
     16       INTEGER DAYS(12)
     17       DATA DAYS/31,28,31,30,31,30,31,31,30,31,30,31/
     18 C
     19       IYR=YY/4
     20       ILEAP=YY-(IYR*4)
     21       IF(ILEAP.EQ.0) DAYS(2)=29
     22       JD=0
     23 C
     24       DO 10 J=1,MM-1
1    25          JD=JD+DAYS(J)
1    26  10      CONTINUE
     27       JD=JD+DD
     28 C
     29       RETURN
     30       END

 Pass One    No Errors Detected
             30 Source Lines
```

```
C-----------------------------------------------------------------------
      SUBROUTINE CLML(US,HF,ML)
C-----------------------------------------------------------------------
C    This subroutine calculates the Monin-Obukhov length.
C
C Input-
C     US: friction velocity (m/s)
C     HF: heat flux (degK*m/s)
C
C Output-
C     ML: Monin-Obukhov length (m)
C
      REAL US,HF,ML
C
      REAL PI,RHO,CP,ALPHA,G,K1,BETA,AUO
      INTEGER MNTM(12),MXTM(12),LNTM
      COMMON/CNSTS/PI,RHO,CP,ALPHA,G,K1,BETA,AUO,
     *             MNTM,MXTM,LNTM
C
      REAL TA,STAB,WIND(2),SIGY(10),AZB(10),LOC(10,3),XQ(10),SIGT,RAIN
      INTEGER YEAR,MONTH,DAY,TIME,CLOUD,NLOC
      LOGICAL LSIGT,LRAIN
      COMMON/METFL/TA,STAB,WIND,SIGY,AZB,LOC,XQ,SIGT,RAIN,
     *             YEAR,MONTH,DAY,TIME,CLOUD,NLOC,LSIGT,LRAIN
C
      IF((TIME.LT.MNTM(MONTH)).OR.(TIME.GT.MXTM(MONTH)).OR.
     *    (STAB.NE.0.AND.STAB.LE.0.06)) GOTO 10
C
C Calculate the Monin-Obukhov length for unstable conditions
C
          ML=-TA*US**3/(G*K1*HF)
          GOTO 20
C
C Calculate the Monin-Obukhov length for nighttime
C
 10       ML=AUO*US**2
C
 20   RETURN
      END

Pass One    No Errors Detected
            39 Source Lines
```

```
 1 C-----------------------------------------------------------------------
 2       SUBROUTINE CLSGY(X,SGY)
 3 C-----------------------------------------------------------------------
 4 C    This subroutine calculates the crosswind standard
 5 C deviations using Brigg's values as recommended by the
 6 C National Working Group on Radionuclide Transport.
 7 C
 8       REAL X,SGY
 9 C
10       REAL TA,STAB,WIND(2),SIGY(10),AZB(10),LOC(10,3),XQ(10),SIGT,RAIN
11       INTEGER YEAR,MONTH,DAY,TIME,CLOUD,NLOC
12       LOGICAL LSIGT,LRAIN
13       COMMON/METFL/TA,STAB,WIND,SIGY,AZB,LOC,XQ,SIGT,RAIN,
14      *             YEAR,MONTH,DAY,TIME,CLOUD,NLOC,LSIGT,LRAIN
15 C
16       IF(.NOT.LSIGT) THEN
17           SGY=(STAB*X)/(SQRT(1.0+0.0001*X))
18       ELSE
19           FX=1/(1.0+SQRT((X/1000)/2.5))
20           SGY=SIGT*FX*X
21       ENDIF
22 C
23       RETURN
24       END

Pass One    No Errors Detected
            24 Source Lines
```

```
      SUBROUTINE CLUB(US,ML,CK1,ZOM,DWN,CRS,ZB,UB)
C-----------------------------------------------------------------------
C    This subroutine calculates mean horizontal velocity.
C
C Input-
C     US: friction velocity (m/s)
C     ML: Monin-Obukhov length (m)
C    CK1: constant
C    ZOM: surface roughness (m)
C    DWN: downwind distance (m)
C    CRS: crosswind distance (m)
C     ZB: mean particle height (m)
C
C Output-
C     UB: mean horizontal velocity (m/s)
C
      REAL US,ML,CK1,ZOM,DWN,CRS,ZB,UB
C
      IF(ZB/ML.GE.0) GOTO 10
          Z=ZB
          IF(DWN.EQ.0.AND.CRS.EQ.0) Z=0
          A=(1-15*(0.6*Z/ML))**0.25
          PSI=2*ALOG((1+A)/2)+ALOG((1+A**2)/2)-2*ATAN(A)
     *        +3.1415927/2
          UB=(US/CK1*(ALOG(0.6*ZB/ZOM)-PSI))
          GOTO 20
C
  10  UB=(US/CK1*(ALOG(0.6*ZB/ZOM)+4.7*ZB/ML))
C
  20  RETURN
      END
```

```
Pass One    No Errors Detected
            32 Source Lines
```

```
C-----------------------------------------------------------------------
      SUBROUTINE CLUS(US,HF)
C-----------------------------------------------------------------------
C    This subroutine calculates the friction velocity
C and heat flux
C
C Output-
C     US: friction velocity (m)
C     HF: heat flux (degK*m/s)
C
C Externals-
C     CLHF(HF)
C
      REAL PI,RHO,CP,ALPHA,G,K1,BETA,AUO
      INTEGER MNTM(12),MXTM(12),LNTM
      COMMON/CNSTS/PI,RHO,CP,ALPHA,G,K1,BETA,AUO,
     *             MNTM,MXTM,LNTM
C
      REAL STELAT,SRCHGT,ZO,ZM,MXDWN
      INTEGER NUS
      LOGICAL USP
      COMMON/STEFL/STELAT,SRCHGT,ZO,ZM,MXDWN,NUS,USP
C
      REAL TA,STAB,WIND(2),SIGY(10),AZB(10),LOC(10,3),XQ(10),SIGT,RAIN
      INTEGER YEAR,MONTH,DAY,TIME,CLOUD,NLOC
      LOGICAL LSIGT,LRAIN
      COMMON/METFL/TA,STAB,WIND,SIGY,AZB,LOC,XQ,SIGT,RAIN,
     *             YEAR,MONTH,DAY,TIME,CLOUD,NLOC,LSIGT,LRAIN
C
      IF((TIME.LT.MNTM(MONTH)).OR.(TIME.GT.MXTM(MONTH)).OR.
     *       (STAB.NE.0.AND.STAB.LE.0.06)) GOTO 10
C
C Calculate the heat flux (necessary in day time)
C
          CALL CLHF(HF)
          IF(HF. LE. 0.0) GOTO 5
          FB=0.105
          IF(ZO/ZM.LE.0.01) FB=0.128+0.005*ALOG(ZO/ZM)
C
C Calculate the friction velocity for day time
C
          USS=K1*WIND(1)/ALOG(ZM/ZO)
          HS=TA*USS**3/(K1*G*ZM)
C
c      write(*,*) 'values from CLUS wind(1), zm,z0,ta,HS,hf,ARG(ln) =',
c     * wind(1), zm,z0,ta,HS,HF,1.0+(1.95+32.6*(ZO/ZM)**0.45)*HF/HS
          US=USS*(1.0+FB*ALOG( 1.0+(1.95+32.6*(ZO/ZM)**0.45)*HF/HS ))
          GOTO 20
C
C Calculate the friction velocity for nighttime
C
```

```
 52   5   write(10,*) ' hf  TIME  month day CLOUD',hf,time,month,day,CLOUD
 53       TIME = MNTM(MONTH) - 1
 54       HF = 0.0
 55 C
 56  10   CONTINUE
 57           U0=SQRT(BETA*ZM/(K1*AU0))
 58           CDN=K1/ALOG(ZM/Z0)
 59           SARG=1-(2*U0/(SQRT(CDN)*WIND(1)))**2
 60           US=CDN*WIND(1)*(0.5+0.5*SQRT(SARG))
 61 C
 62  20   CONTINUE
 63 c      write(*,*) ' RETURN FROM CLUS'
 64       RETURN
 65       END

Pass One    No Errors Detected
            65 Source Lines
```

```
      1 C-----------------------------------------------------------------------
      2       SUBROUTINE CLXQ
      3 C-----------------------------------------------------------------------
      4 C    This subroutine calculates the normalized concentrations.
      5 C
      6 C  Input-
      7 C
      8 C Output-
      9 C     US: friction velocity
     10 C     UB: mean horizontal velocity
     11 C     ML: Monin-Obukhov length
     12 C     HF: surface heat flux
     13 C     XQ(.) and AZQ(.) are the vectors containing the calculated normali
     14 C                      concentrations and the mean paricle height.
     15 C Externals:
     16 C     CLUS(US,HF)
     17 C     CLML(US,HF,ML)
     18 C     CLSGY(X,SGY)
     19 C     CLZB(ML,K1,ZOM,DWN,ZB)
     20 C     FUZB(ML,K1,ZOM,DWN,X)
     21 C     FSZB(ML,K1,ZOM,DWN,X)
     22 C
     23       REAL ML, US, UB(10), HF, HS
     24       COMMON /RESULTS/ ML, US, UB, HF, HS
     25 C
     26       REAL PI,RHO,CP,ALPHA,G,K1,BETA,AUO
     27       INTEGER MNTM(12),MXTM(12),LNTM
     28       COMMON/CNSTS/PI,RHO,CP,ALPHA,G,K1,BETA,AUO,
     29      *             MNTM,MXTM,LNTM
     30 C
     31       REAL STELAT,SRCHGT,ZO,ZM,MXDWN
     32       INTEGER NUS
     33       LOGICAL USP
     34       COMMON/STEFL/STELAT,SRCHGT,ZO,ZM,MXDWN,NUS,USP
     35 C
     36       REAL TA,STAB,WIND(2),SIGY(10),AZB(10),LOC(10,3),XQ(10),SIGT,RAIN
     37       INTEGER YEAR,MONTH,DAY,TIME,CLOUD,NLOC
     38       LOGICAL LSIGT,LRAIN
     39       COMMON/METFL/TA,STAB,WIND,SIGY,AZB,LOC,XQ,SIGT,RAIN,
     40      *             YEAR,MONTH,DAY,TIME,CLOUD,NLOC,LSIGT,LRAIN
     41 C
     42 C Initialize
     43 C
     44 c       write(*,*)  'values from CLXQ wind(1), zm,z0,ta,k1,g =',
     45 c     *              wind(1), zm,z0,ta,k1,g
     46       DO 10 I=1,NLOC
1    47         XQ(I)=0.0
1    48         AZB(I)=0.0
1    49         UB(I) = 0.0
1    50  10    CONTINUE
     51 C
```

```
      C Calculate the friction velocity and heat flux
      C
            CALL CLUS(US,HF)
      C
      C Calculate the Monin-Obukhov length
      C
            CALL CLML(US,HF,ML)
      c      write(*,*) 'return from CLML and will calculate zb'
      C
      C Calculate the virtual distance XO for an elevated source
      C
            IF(SRCHGT.NE.0) THEN
               IF(1/ML.GT.0) THEN
                  XO=FSZB(ML,K1,Z0,0,SRCHGT)
               ELSE
                  XO=FUZB(ML,K1,Z0,0,SRCHGT)
               ENDIF
            ELSE
               XO=0
            ENDIF
      C
            DO 40 I=1,NLOC
1     C
1     C Calculate sigma Y
1     C
1              CALL CLSGY(LOC(I,1),SGY)
1              SIGY(I)=SGY
1     C
1     C Calculate the mean particle height
1     C
1              CALL CLZB(ML,K1,Z0,LOC(I,1)+XO,ZB)
1              AZB(I)=ZB
1              IF(ZB.LT.0) GOTO 40
1     C
1     C Calculate the mean horizontal velocity
1     C
1              CALL CLUB(US,ML,K1,Z0,LOC(I,1),LOC(I,2),ZB,UB1)
1              UB(I) = UB1
1     C
1     C Calculate the dilution factor
1     C
1              DN=SQRT(2*PI)*UB1*ZB*SIGY(I)
1              CRS=ABS(LOC(I,2))
1              EARG1=-(0.66*LOC(I,3)/ZB)**1.5
1              EARG2=CRS**2/(2*SIGY(I)**2)
1              XO(I)=0.73/DN*EXP(EARG1-EARG2)
1     40    CONTINUE
            RETURN
            END

Pass One    No Errors Detected
```

100 Source Lines

```
C-----------------------------------------------------------------------
      SUBROUTINE CLZB(ML,CK1,ZOM,DWN,ZB)
C-----------------------------------------------------------------------
C    This subroutine calculates the mean particle height
C
C Input-
C     ML: Monin-Obukhov length (m)
C
C Output-
C     ZB: mean particle heights for each downwind distance (m)
C    CK1: von Karman constant
C    ZOM: surface roughness (m)
C    DWN: downwind distance (m)
C
C Externals-
C     FUZB(ML,CK1,ZOM,DWN,X)
C     FSZB(ML,CK1,ZOM,DWN,X)
C
      REAL ML,CK1,ZOM,DWN,ZB
C
C Solve for ZB
C
      ZB=-1.0
      XTOL=0.0001
      A=0.0001
      B=4000.0
      IFLAG=0
      N=1
C
c      write(*,*) ' enter CLZB '
      IF(A/ML.LT.0) THEN
          FA=FUZB(ML,CK1,ZOM,DWN,A)
      ELSE
          FA=FSZB(ML,CK1,ZOM,DWN,A)
      ENDIF
C
C Check for sign change
C
      IF(B/ML.LT.0) THEN
          FB=FUZB(ML,CK1,ZOM,DWN,B)
      ELSE
          FB=FSZB(ML,CK1,ZOM,DWN,B)
      ENDIF
      IF(FA*FB.LE.0.0) GOTO 10
          IFLAG=2
          GOTO 50
  10  ERROR=ABS(B-A)
  20  ERROR=ERROR/2
C
C Check for sufficiently small interval
C
```

```
      IF(ERROR.LT.XTOL) GOTO 50
      XM=(A+B)/2
C
C Check for unreasonable error requirement
C
      IF(XM+ERROR.EQ.XM) GOTO 40
          IF(XM/ML.LT.0) THEN
              FM=FUZB(ML,CK1,ZOM,DWN,XM)
          ELSE
              FM=FSZB(ML,CK1,ZOM,DWN,XM)
          ENDIF
C
      N=N+1
C
C Change to a new interval
C
      IF(FA*FM.LE.0) GOTO 30
          A=XM
          FA=FM
          GOTO 20
C
 30   B=XM
      GOTO 20
C
 40   IFLAG=1
C
 50   IF(IFLAG.NE.2) ZB=(A+B)/2
C
c      write(*,*) ' exit CLZB'
      RETURN
      END
```

```
Pass One    No Errors Detected
            82 Source Lines
```

```
      1 C-----------------------------------------------------------------------
      2       FUNCTION DISC(RND, N, XV, FX)
      3 C-----------------------------------------------------------------------
      4 C This function generate discrete random value with c.d.f given by FX
      5 C
      6 C Input value -
      7 C       RND : a random number between 0 - 1
      8 C         N : number of diferent values of X
      9 C        XV : the vector of values
     10 C        FX : the vector with the cumulative distribution function of XV
     11 C
     12 C Output value -
     13 C      DISC : a random valuewith the P.D.F given by FX
     14 C
     15       DIMENSION XV(N), FX(N)
     16       IF (RND . LT. FX(1)) THEN
     17          DISC = XV(1)
     18          RETURN
     19       ELSE
     20          DO 10 I = 1,N-1
1    21          IF((RND. GT . FX(I)) .AND. (RND. LE. FX(I+1))) DISC = XV(I+1)
1    22  10      CONTINUE
     23       ENDIF
     24       RETURN
     25       END

 Pass One    No Errors Detected
             25 Source Lines
```

```
C-----------------------------------------------------------------------
      FUNCTION FSZB(ML,CK1,ZOM,DWN,X)
C-----------------------------------------------------------------------
C    This function calculates mean particle height for stable
C conditions.
C
C Input-
C     ML: Monin-Obukhov length (m)
C    CK1: von Karman constant
C    ZOM: surface roughness (m)
C    DWN: downwind distance (m)
C      X: mean particle height (m)
C
      REAL ML,CK1,ZOM,DWN,X
C
      FSZB=(0.74/CK1**2)*X*((ALOG(0.6*X/ZOM)+4.9*X/ML)*
     *   (1+4.9*X/ML)+1.2*X/ML)-DWN
C
      RETURN
      END

Pass One    No Errors Detected
            20 Source Lines
```

```
 1 C-------------------------------------------------------------------
 2       FUNCTION FUZB(ML,CK1,ZOM,DWN,X)
 3 C-------------------------------------------------------------------
 4 C    This function calculates mean particle height for
 5 C for unstable conditions.
 6 C
 7 C Input-
 8 C     ML: Monin-Obukhov length (m)
 9 C    CK1: constant
10 C    ZOM: surface roughness (m)
11 C    DWN: downwind distance (m)
12 C      X: mean particle height (m)
13 C
14       REAL ML,CK1,ZOM,DWN,X
15 C
16       A=(1-15*(0.6*X/ML))**0.25
17       PSI=2*ALOG((1+A)/2)+ALOG((1+A**2)/2)-2*ATAN(A)+3.1415927/2
18       FUZB=(0.74/CK1**2)*X*((ALOG(0.6*X/ZOM)-PSI)/
19      *   (1-3.5*X/ML)**0.5)-DWN
20 C
21       RETURN
22       END

Pass One    No Errors Detected
            22 Source Lines
```

```
C-----------------------------------------------------------------------
      SUBROUTINE GENRAN(STELAT,SRCHGT,ZO,ZM,TA,STAB,WIND,XLOC,
     *                  SIGT,YEAR,MONTH,DAY,TIME,CLOUD)
C-----------------------------------------------------------------------
C
C  This routine generates random values for the input of the CRDM
C
C  The distributions are:
C     STELAT: fixed at 45 degrees
C     SRCHGT: binary at 0 m or at 20 m with the same probability
C         ZO: uniform (.1,1.0)
C         ZM: fixed at 10 m
C         TA: date dependent (see the table for each season)
C       STAB: Not generated (instead use SIGT)
C       WIND: season dependent (see table - Note it is assumed wind > 1.
C       XLOC: uniform (500,10000)
C       SIGT: uniform (2.1,22.5)
C       YEAR: fixed 1985
C      MONTH: uniform (1,12)
C        DAY: month dependent
C       TIME: uniform (000,2400)  taken only clock values
C      CLOUD: uniform (0-8) only integer values
C
C External Routines and/or Functions
C     FUNCTION DISC(RND, N, XV, FX)
C     FUNCTION UNIF(RND, A, B)
C     FUNCTION Z12(RND, X1, X2, P)
C     FUNCTION RANDOM(L)
C
      REAL XTA(2,4),PWIND(6,4),XWIND(7),XSIGT(2),XZO(2),XSRCHGT(2),
     *     XWIND1(6)
      INTEGER YEAR,XMONTH(12),XDAY(12),TIME,XCLOUD(8),DAY,MONTH,CLOUD
      COMMON /RAND/ IX,IY,IZ
C
C     DATA PWIND/.2083,.6666,.8981,.9763,.9967,1.000,1.000,
C    *           .3442,.7315,.9256,.9741,.9972,.9995,1.000,
C    *           .5288,.9318,.9954,.9996,1.000,1.000,1.000,
C    *           .4084,.8342,.9594,.9903,.9995,1.000,1.000/,
      DATA  XTA/-15.,1.0,0.0,15.0,15.0,30.0,0.0,12.0/,
     *      XWIND/2.778,5.556,8.3333,11.111,13.8889,16.667,20.000/
     *      XWIND1/1.0,2.0,3.0,4.0,5.0,6.0/,
     *      PWIND/.5789,.8713,.9701,.9958,1.000,1.000,
     *            .5906,.8866,.9605,.9956,.9992,1.000,
     *            .8553,.9902,.9992,1.000,1.000,1.000,
     *            .4258,.9314,.9836,.9992,1.000,1.000/,
     *      XDAY/31,28,31,30,31,30,31,31,30,31,30,31/,
     *      XZO/.10,1.00/,XSIGT/2.1,22.5/,XSRCHGT/0.0,20.0/,
     *     XWIND/7/
C     ZM = 10.0
C     STLAT = 45.0
C     YEAR = 1985
```

```
C
C Generate the value of date: MONTH and DAY
C
      RND = RANDOM(L)
      MONTH = INT( UNIF(RND,1.0,13.0) )
      RND = RANDOM(L)
      DAY = INT( UNIF(RND,1.0, FLOAT( XDAY(MONTH)+1 )) )
      RND = RANDOM(L)
      TIME = INT( UNIF(RND, 0.0, 24.)*100.)
      IHR = 100 * INT(TIME/100)
      TIME = IHR + INT( 0.6*FLOAT(TIME-IHR) )
C
C Generate the value of temperature
C
      RND = RANDOM(L)
      IF((MONTH.EQ.12).OR.(MONTH.EQ.1).OR.(MONTH.EQ.2)) ISEA = 1
      IF((MONTH.EQ.3).OR.(MONTH.EQ.4).OR.(MONTH.EQ.5)) ISEA = 2
      IF((MONTH.EQ.6).OR.(MONTH.EQ.7).OR.(MONTH.EQ.8)) ISEA = 3
      IF((MONTH.EQ.9).OR.(MONTH.EQ.10).OR.(MONTH.EQ.11)) ISEA = 4
      TA = ANINT( UNIF(RND, XTA(1,ISEA), XTA(2,ISEA)) )+273.16
C
C Generate the value of wind speed
C
      RND = RANDOM(L)
      IVAL = INT( DISC(RND, 6, XWIND1, PWIND(1,ISEA)) )
      RND = RANDOM(L)
      WIND = UNIF(RND, XWIND(IVAL), XWIND(IVAL+1))
C
C Generate cloud cover variable
C
      RND = RANDOM(L)
      CLOUD = INT( UNIF(RND, 0.0, 9.0) )
C
C Generate Z0 variable
C
      RND = RANDOM(L)
      Z0 = UNIF(RND, XZ0(1), XZ0(2))
C
C Generate SRCHGT variable
C
      RND = RANDOM(L)
      SRCHGT = Z12(RND, XSRCHGT(1), XSRCHGT(2),0.5)
C
C Generate LOC variable
C
      RND = RANDOM(L)
      XLOC = INT( UNIF(RND, 500.0, 10000.0) )
C
C Generate SIGT variable
C
      RND = RANDOM(L)
```

```
  103       SIGT = UNIF(RND, XSIGT(1), XSIGT(2))
  104 C
  105       RETURN
  106       END

Pass One    No Errors Detected
            106 Source Lines
```

```
 1 C-----------------------------------------------------------------------
 2       SUBROUTINE JDAY(IYEAR, NDAY, MYY, MM, MDD)
 3 C-----------------------------------------------------------------------
 4 C THIS PROGRAM CALCULATES THE JULIAN DAY CORRESPONDING TO THE NUMBER OF
 5 C DAYS SINCE JAN 1 4713, B.C.
 6 C
 7 C   INPUT DATA:
 8 C    IYEAR : YEAR
 9 C     NDAY : NUMBER OF DAYS SINCE JAN 1 OF THE YEAR
10 C
11 C   OUPUT DATA:
12 C     JULIAN DATE ON (IYEAR,NDAY)
13 C     MYY, MM, MDD
14 C
15 C----CALCULATE THE JULIAN DATE FOR JAN 1 OF YEAR
16 C
17       JD = INT(365.25*(IYEAR-1)) + INT(30.6001*14)
18       JD = NDAY + JD
19 C
20 C----NOW CALCULATE THE DATE
21 C
22       MYY = INT( (JD-122.1)/365.25 )
23       MM = INT( (JD - INT(365.25*MYY))/30.6001)
24       MDD = JD - INT(365.25*MYY) - INT(30.6001*MM)
25       IF(MM.GE.14) THEN
26          MM = MM - 13
27       ELSE
28          MM = MM - 1
29       ENDIF
30       IF(MM .EQ. 1. OR. MM. EQ. 2) MYY = MYY + 1
31       RETURN
32       END

Pass One    No Errors Detected
            32 Source Lines
```

```
 1 C-----------------------------------------------------------------------
 2       FUNCTION RANDOM(L)
 3 C-----------------------------------------------------------------------
 4 C        ALGORITHM AS 183  APPL. STATIST. (1982) VOL.31, NO.2
 5 C        RETURNS A PSEUDO-RANDOM NUMBER RECTANGULARLY DISTRIBUTED
 6 C        BETWEEN 0 AND 1
 7 C
 8 C        IX, IY AND IZ SHOULD BE SET TO INTEGER VALUES BETWEEN
 9 C        1 AND 30000 BEFORE FIRST ENTRY
10 C
11 C        INTEGER ARITHMETIC UP TO 30323 IS REQUIRED
12 C
13       COMMON /RAND/ IX, IY, IZ
14       IX = 171 * MOD(IX, 177) - 2  * (IX / 177)
15       IY = 172 * MOD(IY, 176) - 35 * (IY / 176)
16       IZ = 170 * MOD(IZ, 178) - 63 * (IZ / 178)
17 C
18       IF (IX .LT. 0) IX = IX + 30269
19       IF (IY .LT. 0) IY = IY + 30307
20       IF (IZ .LT. 0) IZ = IZ + 30323
21       RANDOM = AMOD(FLOAT(IX) / 30269.0 + FLOAT(IY) / 30307.0 +
22      *              FLOAT(IZ) / 30323.0, 1.0)
23       IF(RANDOM. GT. 0.0) RETURN
24       RANDOM = DMOD( DBLE(FLOAT(IX))/30296.000 +
25      *     DBLE(FLOAT(IY))/30307.000 + DBLE(FLOAT(IZ))/30323.000,1.000)
26       IF (RANDOM. GE. 1.0) RANDOM = 0.999999
27       RETURN
28       END

Pass One    No Errors Detected
            28 Source Lines
```

```
C-----------------------------------------------------------------------
      SUBROUTINE RDCAS(NLOC,LOC,VYEAR,VMONTH,VDAY,VTIME,VCLOUD,VSTELAT,
     *         VSRCHGT,VZO,VZM
     *        ,NYEAR,NMONTH,NDAY,NTIME,NCLOUD,NSTELAT,NSRCHGT,NZO,NZM
     *        ,VTA,VWIND,VSIGT, NTA,NWIND,NSIGT,LGUTI, LGUTO,LGUO,FNAME)
C-----------------------------------------------------------------------
C This routine read the 'grid' of the sample space
C
C Input data -
C       LGUTI: logical terminal input number
C       LGUTO: logical terminal output number
C        LGUO: logical file output number
C
C  Output data -
C         LOC: Vector of values of location coordinates
C        NLOC: number of locations
C       VYEAR: Vector of values of the year at time date IDATE
C      VMONTH: Vector of values of the months at time date IDATE
C        VDAY: Vector of values of the day at time date IDATE
C       VTIME: Vector of values of the time
C      VCLOUD: Vector of values of the cloud cover
C     VSTELAT: Vector of values of the source latitudes
C     VSRCHGT: Vector of values of the the source height
C         VZO: Vector of values of the source roughness
C         VZM: Vector of values of the measurement height
C       NYEAR: Equal to IDATE
C      NMONTH: Equal to IDATE
C        NDAY: Equal to IDATE
C       NTIME: Number of different times
C      NCLOUD: Number of different cloud cover values
C     NSTELAT: Number of different source latitudes
C     NSRCHGT: Number of different source height
C         NZO: Number of different source roughness
C         NZM: Number of different measurements height
C         VTA: Vector of values of temperature
C       VWIND: Vector of values of wind speed
C       VSIGT: Vector of values of sigma theta
C         NTA: number of different temperatures
C       NWIND: number of different wind speed values
C       NSIGT: number of different sigma theta values
C
C Note: the temperature values are assumed to depend upon the DATE
C
      REAL VTA(3,3),VWIND(3),VSIGT(3),LOC(10,3)
      INTEGER VYEAR(3),VMONTH(3),VDAY(3),VTIME(3),VCLOUD(3)
      REAL VSTELAT(3),VSRCHGT(3),VZO(3),VZM(3)
      CHARACTER FNAME*80, YES*1
      LOGICAL IFEXT,IFSRH
C
C  For each variable read the number of points and the values of the poi
C
```

```
      LGUT = LGUTI
      WRITE(LGUTO, *) ' WANT TO ENTER INTERACTIVE INPUT?'
      READ(LGUT, 5) YES
      IF(YES. EQ.'Y'. OR . YES .EQ. 'y') THEN
         IFSRH = .TRUE.
      ELSE
         LGUTI = 2
         IFSRH = .FALSE.
         WRITE(LGUTO, *) ' ENTER INPUT FILE NAME'
         READ(LGUT, 5) FNAME
 5       FORMAT(A)
         OPEN(UNIT=LGUTI,FILE =FNAME)
      ENDIF
C
      IF(IFSRH) WRITE(LGUTO, *) ' NUMBER OF LOCATIONS: NLOC'
      READ(LGUTI, *) NLOC
      NLOC = MIN0(NLOC,10)
      DO 20 ILOC=1,NLOC
1        IF(IFSRH) WRITE(LGUTO,  10) ILOC
1  10    FORMAT(' ENTER COORDENATES FOR LOCATION: ',I2,' (dwd,crs,z)')
1        READ(LGUTI, *) (LOC(ILOC,J),J = 1,3)
1  20 CONTINUE
C
      IF(IFSRH) WRITE(LGUTO, *) 'NUMBER OF DATES: NDATE'
      READ(LGUTI, *) NDATE
      NDATE = MIN0(NDATE,3)
      NYEAR = NDATE
      NMONTH = NDATE
      NDAY = NDATE
      DO 40 IDATE=1,NDATE
1        IF(IFSRH) WRITE(LGUTO, 30) IDATE
1  30    FORMAT(' ENTER DATE NUMBER: ',I2,' YEAR(YYYY) MONTH(MM) ',
1     *           'DAY(DD)')
1        READ(LGUTI, *) VYEAR(IDATE), VMONTH(IDATE), VDAY(IDATE)
1  40 CONTINUE
C
      IF(IFSRH) WRITE(LGUTO, *) ' NUMBER OF TIMES OF DAY: NTIME'
      READ(LGUTI, *) NTIME
      NTIME = MIN0(NTIME,3)
      DO 60 ITIME=1,NTIME
1        IF(IFSRH) WRITE(LGUTO, 50) ITIME
1  50    FORMAT(' ENTER TIME NUMBER: ',I2,'  as (HHMM)')
1        READ(LGUTI, *) VTIME(ITIME)
1  60 CONTINUE
C
      IF(IFSRH) WRITE(LGUTO, *) ' NUMBER OF CLOUD COVER VALUES: NCLOUD'
      READ(LGUTI, *) NCLOUD
      NCLOUD = MIN0(NCLOUD,3)
      DO 80 ICLOUD=1,NCLOUD
1        IF(IFSRH) WRITE(LGUTO, 70) ICLOUD
1  70    FORMAT(' ENTER CLOUD VALUE NUMBER: ',I2,'  (0 <= CLOUD<= 8)')
```

```
1   103          READ(LGUTI, *) VCLOUD(ICLOUD)
1   104  80    CONTINUE
    105 C
    106        IF(IFSRH) WRITE(LGUTO, *)
    107       *          ' NUMBER OF SOURCE LATITUD VALUES: NSTELAT'
    108        READ(LGUTI, *) NSTELAT
    109        NSTELAT = MINO(NSTELAT,3)
    110        DO 100 ISTELAT=1,NSTELAT
1   111           IF(IFSRH) WRITE(LGUTO, 90) ISTELAT
1   112  90       FORMAT(' ENTER SOURCE LATITUD VALUE NUMBER: ',I2,'  (Degrees)')
1   113           READ(LGUTI, *) VSTELAT(ISTELAT)
1   114  100   CONTINUE
    115 C
    116        IF(IFSRH) WRITE(LGUTO, *)
    117       *         ' NUMBER OF SOURCE HEIGHT VALUES: NSRCHGT'
    118        READ(LGUTI, *) NSRCHGT
    119        NSRCHGT = MINO(NSRCHGT,3)
    120        DO 120 ISRCHGT=1,NSRCHGT
1   121           IF(IFSRH) WRITE(LGUTO, 110) ISRCHGT
1   122  110      FORMAT(' ENTER SOURCE HEIGHT VALUE NUMBER: ',I2,'  in m')
1   123           READ(LGUTI, *) VSRCHGT(ISRCHGT)
1   124  120   CONTINUE
    125 C
    126        IF(IFSRH) WRITE(LGUTO, *)
    127       *         ' NUMBER OF SURFACE ROUGHNESS VALUES: NZO'
    128        READ(LGUTI, *) NZO
    129        NZO = MINO(NZO,3)
    130        DO 140 IZO=1,NZO
1   131           IF(IFSRH) WRITE(LGUTO, 130) IZO
1   132  130      FORMAT(' ENTER SURFACE ROUGHNESS VALUE NUMBER: ',I2,'  in m')
1   133           READ(LGUTI, *) VZO(IZO)
1   134  140   CONTINUE
    135 C
    136        IF(IFSRH) WRITE(LGUTO, *)
    137       *          ' NUMBER OF MEASUREMENT HEIGHT VALUES: NZM'
    138        READ(LGUTI, *) NZM
    139        NZM = MINO(NZM,3)
    140        DO 160 IZM=1,NZM
1   141           IF(IFSRH) WRITE(LGUTO, 150) IZM
1   142  150      FORMAT(' ENTER MEASUREMENT HEIGHT VALUE NUMBER: ',I2,'  in m')
1   143           READ(LGUTI, *) VZM(IZM)
1   144  160   CONTINUE
    145 C
    146        IF(IFSRH) WRITE(LGUTO, *) ' NUMBER OF TEMPERATURE VALUES: NTA'
    147        READ(LGUTI, *) NTA
    148        NTA = MINO(NTA,3)
    149        DO 180 IDATE=1,NDATE
1   150        DO 180 ITA=1,NTA
2   151           IF(IFSRH) WRITE(LGUTO, 170)
2   152       *            ITA,VYEAR(IDATE), VMONTH(IDATE),VDAY(IDATE)
2   153  170      FORMAT(' ENTER TEMPERATURE VALUE NUMBER: ',I2,' For date ',
```

```
2   154      *          I4,2(1X,I2))
2   155         READ(LGUTI, *) VTA(IDATE,ITA)
2   156         VTA(IDATE,ITA) = VTA(IDATE,ITA) + 273.16
2   157  180  CONTINUE
    158 C
    159       IF(IFSRH) WRITE(LGUTO, *) ' NUMBER OF WIND SPEED VALUES: NWIND'
    160       READ(LGUTI, *) NWIND
    161       NWIND = MINO(NWIND,3)
    162       DO 200 IWIND=1,NWIND
1   163          IF(IFSRH) WRITE(LGUTO, 190) IWIND
1   164  190     FORMAT(' ENTER WIND SPEED VALUE NUMBER: ',I2,'   in m/s')
1   165          READ(LGUTI, *) VWIND(IWIND)
1   166  200  CONTINUE
    167 C
    168       IF(IFSRH) WRITE(LGUTO, *) ' NUMBER OF SIGMA THETA: NSIGT'
    169       READ(LGUTI, *) NSIGT
    170       NSIGT = MINO(NSIGT,3)
    171       DO 220 ISIGT=1,NSIGT
1   172          IF(IFSRH) WRITE(LGUTO, 210) ISIGT
1   173  210     FORMAT(' ENTER SIGMA THEYA VALUE NUMBER: ',I2 )
1   174          READ(LGUTI, *) VSIGT(ISIGT)
1   175  220  CONTINUE
    176 C
    177       IF(.NOT.IFSRH)  CLOSE(UNIT=LGUTI)
    178       IF(.NOT.IFSRH)  LGUTI = LGUT
    179       RETURN
    180       END

 Pass One    No Errors Detected
            180 Source Lines
```

```
C-----------------------------------------------------------------------
      SUBROUTINE SIMVAL(STELAT,SRCHGT,ZO,ZM,TA,STAB,WIND,XLOC,
     *    SIGT,YEAR,MONTH,DAY,TIME,CLOUD,LGUF,IFEND)
C-----------------------------------------------------------------------
C
C  This routine read the values for the input of the CRDM obtained
C  from the metereological Tape and generate the other values.
C
C  The distributions are:
C     STELAT: fixed at 45 degrees
C     SRCHGT: binary at 0 m or at 20 m with the same probability
C         ZO: uniform (.1,1.0)
C         ZM: fixed at 10 m
C         TA: date dependent (average of the month at Port Hope)
C       STAB: read from the TAPE
C       WIND: read from the TAPE
C       XLOC: uniform (500,10000)
C       SIGT: depend on stability class read from the TAPE
C       YEAR: read from the TAPE
C      MONTH: obtained from NDAY (read from the TAPE)
C        DAY: obtained from NDAY (read from the TAPE)
C       TIME: read from the TAPE
C      CLOUD: obtained from STAB, a method based on the monograph given
C              in Pasquill ('Atmospheric Dispersion' 2nd Edition, 1984,
C              Ellis Harwood) and a simulation method
C
C External Routines and/or Functions
C     FUNCTION DISC(RND, N, XV, FX)
C     FUNCTION UNIF(RND, A, B)
C     FUNCTION Z12(RND, X1, X2, P)
C     FUNCTION RANDOM(L)
C     SUBROUTINE JDAY(IYEAR, NDAY, MYY, MM, MDD)
C     SUBROUTINE SOLRAD(HF, WIND, STAB)
C
      DIMENSION TEMP(12), SIGMA(7)
      CHARACTER FRMRF*30
      REAL XZO(2),XSRCHGT(2)
      INTEGER YEAR,TIME,CLOUD,DAY,MONTH
      COMMON /RAND/ IX,IY,IZ
C
      REAL PI,RHO,CP,ALPHA,G,K1,BETA,AUO
      INTEGER MNTH(12),MXTH(12),LNTH
      COMMON/CNSTS/PI,RHO,CP,ALPHA,G,K1,BETA,AUO,
     *            MNTH,MXTH,LNTH
C
      REAL RF(9)
      DATA RF/1.0,0.89,0.81,0.76,0.72,0.67,0.59,0.45,0.23/
      DATA  XZO/.10,1.00/, XSRCHGT/0.0,20.0/
C
      DATA TEMP/-5.8, -5.1, -0.5, 6.0, 11.3, 16.1,
     1          19.5, 19.5, 15.6,  9.5, 3.8, -3.1/
```

```
      DATA SIGMA/22.5, 20.0, 15.0, 10.0, 5.65, 2.95, 2.1/
C
C Generate Z0 variable
C
      RND = RANDOM(L)
      Z0 = UNIF(RND, XZ0(1), XZ0(2))
C
C Generate SRCHGT variable
C
      RND = RANDOM(L)
      SRCHGT = Z12(RND, XSRCHGT(1), XSRCHGT(2),0.5)
C
C Generate LOC variable
C
      RND = RANDOM(L)
      XLOC = INT( UNIF(RND, 500.0, 10000.0) )
C
C  Read TAPE
C
      FRMRF = '(1X,I4,1X,I3,1X,I2,1X,3I2,I3)'
 10   READ(LGUF,FRMRF,END=1000) NY, NDAY, NHH, NMD, NWS ,NSC, NPRE
C     write(10,9) NY, NDAY, NHH, NMD, NWS ,NSC, NPRE
C 9   FORMAT('NY,NDAY,NHH,NMD,NWS,NSC,NPRE',10X,I4,1X,I3,1X,I2,1X,
C    1        3(I2,1X),I3)'
C
C Obtain value of WIND
C
      NC = NSC/10
      NCLASS = NSC - (10*NC)
      WIND = FLOAT(NWS) + (0.1*FLOAT(NC))
      IF(WIND. LT. 2.7) GOTO 10
C
C Obtain TIME, DAY, MONTH, YEAR
C
      CALL JDAY(NY, NDAY, YEAR, MONTH, DAY)
      TIME = 100*NHH
C
C  Obtain  values of TA and SIGT
C
      TA = TEMP(MONTH) + 273.16
      SIGT = SIGMA(NCLASS)
C
C  Calculate value of CLOUD . If it is night time or hf=0, unifirmly
C  generates the value of CLOUD
C
      IF((NCLASS.GT.4) .OR. (TIME.LT.MNTH(MONTH))
     1                 .OR. (TIME.GE.MXTH(MONTH))) THEN
         hf = -1.0
         r = -1.0
         RND = RANDOM(L)
         CLOUD = INT( UNIF(RND, 0.0, 9.0) )
```

```
      103       ELSE
      104          CALL SOLRAD(HF, WIND, NCLASS)
      105          PHI=ATAN(0.4348*SIN(PI*(FLOAT(NDAY)-78.0)/180.0))
      106          TAU=PI*(TIME-LNTM)/1200.
      107          RLAT=STELAT*PI/180.
      108          H0=950*(COS(PHI)*COS(TAU)*COS(RLAT)+SIN(PHI)*SIN(RLAT))
      109          R = HF/H0
      110          DO 20 I = 1,9
    1 111             IF(R .GE. RF(I) ) GOTO 30
    1 112  20       CONTINUE
      113          I = I - 1
      114  30       CLOUD = I - 1
      115       ENDIF
      116 c     write(10,40) year,month ,day,time, ta, wind,nclass, cloud,hf,h0,r
      117 c40   format(/' year month day time ta     wind class cloud hf  h0  r'/
      118 c    1   1x,i4,3x,i2,3x,i2,1x,i4,1x,f6.2,1x,f4.1,2(3x,i2),3(2x,f5.1))
      119 C
      120       RETURN
      121  1000 IFEND = 1
      122       RETURN
      123       END

 Pass One    No Errors Detected
            123 Source Lines
```

```
 1 C-----------------------------------------------------------------------
 2       SUBROUTINE SOLRAD(HF, WIND, NCLASS)
 3 C-----------------------------------------------------------------------
 4 C    This subroutine calculates the CLOUD COVER from heat flux.
 5 C    The method is based on the monograph given in Pasquill ('Atmospheri
 6 C    Dispersion', 2nd. edition, (1984), Ellis Harwood), and a simulation
 7 C    method. It uses the stability class and the wind direction to obtai
 8 C    the range of values the heat flux could take. Then uses a random
 9 C    realization within that range.
10 C
11 C Output-
12 C     HF: heat flux (degK*m/s)
13 C
14 C Externals-
15 C     RND
16 C
17       REAL HF
18       COMMON /RAND/ IX, IY, IZ
19 C
20       REAL RG(9,4)
21       DATA RG/480.0,540.0,600.0,730.0,890.0,890.0,890.0,890.0,890.0,
22      2        270.0,300.0,320.0,360.0,450.0,690.0,890.0,890.0,890.0,
23      3        130.0,135.0,140.0,160.0,180.0,230.0,400.0,600.0,890.0,
24      4         20.0, 20.0, 20.0, 20.0, 20.0, 20.0, 20.0, 20.0, 20.0/
25 C
26       XMAX = 890.0
27       ICL = NCLASS
28       IF (ICL. GT. 4) THEN
29          HF = 0.0
30       ELSE
31          IWND = INT(WIND+0.5) + 1
32          IF(IWND. GT. 9) IWND = 9
33          RND = RANDOM(L)
34          F1 = RG(IWND,ICL)
35          F2 = 890.0
36          IF(ICL .GT. 1) F2 = RG(IWND,ICL-1)
37          HF = UNIF(RND,F1,F2)
38 C       write(10,11) hf, f1, f2 , rnd,ICL, IWND
39 C 11    FORMAT(//' SOLRAD: hf,f1,f2,rnd,icl,iwnd ' ,4(1X,E9.3),2I3)
40       ENDIF
41 C
42       RETURN
43       END

Pass One    No Errors Detected
            43 Source Lines
```

```
 1 C-----------------------------------------------------------------------
 2       FUNCTION UNIF(RND, A, B)
 3 C-----------------------------------------------------------------------
 4 C This function generate a UNIFORM random value distributed
 5 C between A and B
 6 C
 7 C Input value -
 8 C       RND : a random number between 0 - 1
 9 C      A, B : the the range of possible values
10 C
11 C Output value -
12 C       UNIF : a random number with UNIFORM distribution between (a,b)
13 C
14       RI = ABS(B - A)
15       UNIF = RI*RND + AMIN1(A,B)
16       RETURN
17       END

Pass One    No Errors Detected
            17 Source Lines
```

```
 1 C-----------------------------------------------------------------------
 2       FUNCTION Z12(RND, X1, X2, P)
 3 C-----------------------------------------------------------------------
 4 C This function generate a random value with two possible values: X1
 5 C with probability P and X2 with probability 1 - P
 6 C
 7 C Input value -
 8 C       RND : a random number between 0 - 1
 9 C X1 and X2 : the two possible values
10 C         P : probability of X1
11 C
12 C Output value -
13 C       Z12 : a random number with the distribution of Z0
14 C
15       Z12 = X2
16       IF(RND. LE. P) Z12 = X1
17       RETURN
18       END

Pass One    No Errors Detected
            18 Source Lines
```

DISTRIBUTION

* Dr. D. Mills	Director of Research
Dr. S. Daggupaty	Environment Canada 4905 Dufferin St, Downsview, Ont M3H 5T4
Dr. P.K. Misra	Ministry of the Environment Air Resources Branch, 125 Resources Road East Wing, Rexdale, Ont M9W 5L1
Mr. P.M. Foster	Central Electricity Research Laboratories Applied Physics Div, Environmental Section CEGB, Kelvin Ave, Leatherhead, KT22 & SE Surrey, UK
Librarian	Check List & Cataloguing Services, Ontario Government, 180 Bloor Street, West, 5th Floor Toronto, Ontario
Mr. E.M. Laratta	Design & Development - Gen, H10E02
Mr. C.A. Morrison	Design & Development - Gen, H11E22
* Mr. R.J. Fluke	Design & Development - Gen, H11D21
* Mr. A.E. Boyer	Power Systems Operations Div, A06E01
Mr. B. Neil	SSD, Pickering
Mr. F. Vanderzwet	Bruce NGS 'B'
Mr. M.W. White	Teach & Train Serv, AA2H01
Mr. R.W. Maruska	Tech & Train Serv, A02C06
Dr. G.L. Ogram	Chemical Research, KD138
* Mr. G.C. Edwards	Chemical Research, KD138
* Mr. Y.T. Tam	Chemical Research, KD138
* Mr. O.T. Melo	Chemical Research, KD138
Mrs. S.C. Wright	Chemical Research, KD138
* Dr. J. Endrenyi	Operations Research, KR297
* Dr. A.L. Punhani	Operations Research, KR297
Dr. F. Camacho	Operations Research, KR297

Public Reference Centre
Editorial Section (2)
Office Copies (3)
Extra Copies (3)

* Do not receive Appendices. Include covering memo.